TimePod®

5330A Programmable Cross Spectrum Analyzer

Operation and Service

Revision 1.2

May 28, 2013

milesdesign

Manual and software copyright © 2012 Miles Design LLC. All rights reserved.

TimePod® is a registered trademark of Symmetricom, Inc.

16758 45th Ave NE Lake Forest Park, WA 98155 425-351-5226 www.miles.io

Support: john@miles.io

Table of Contents

Introduction 9

Specifications 11

Getting started 13

- What's in the box? 13
- Sales and technical support 13
- USB driver and software installation 15
- Front panel features 17
- Rear panel features 19
- Choosing an external reference 20
- Making your first measurements 21
- Tips for new users 24
- A brief architectural note 27

Making measurements 29

- "What's all this ADEV stuff, anyhow?" 31
  - Allan Deviation (a) 32
  - Modified Allan Deviation (m) 32
  - Hadamard Deviation (h) 33
  - Time Deviation (t) 33
  - Examining changes in stability over time 34
  - Common artifacts in ADEV and related measurements 35
  - Hints for xDEV measurements 40
- Working with phase- and frequency-difference traces 43
  - Phase difference (Original) (w) 43
  - Phase difference (Unwrapped) (p) 43
  - Frequency difference (f) 44
  - How does TimeLab measure frequency? 45
  - Phase/frequency measurements with the TimePod 45
  - Measurement initialization 46
  - Examining traces in detail 48

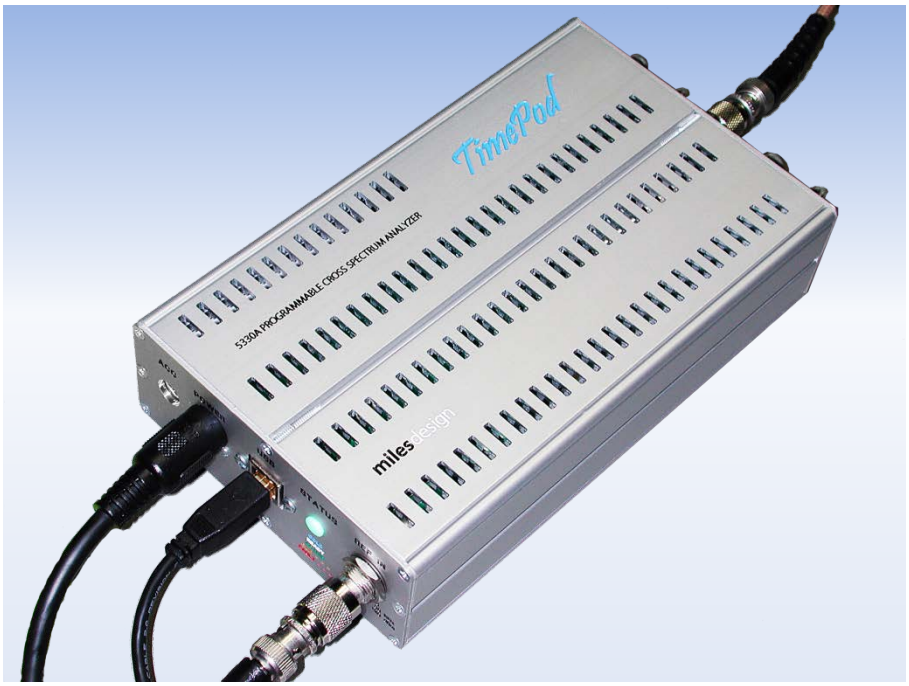
Navigating zoomed graphs 49
Hints for phase/frequency stability measurements 51
Phase noise, AM noise, and jitter 53
Integrated noise and jitter measurement 54
The spur table 55
Show or hide known spurs 55
Spur measurement options 56
Is it a spur, or isn't it? 57
Understanding instrument spurs 57
Hints for noise measurements 59
TimeLab command reference 61
File menu 63
Load .TIM file (l) 63
Save image or .TIM file (s) 63
Copy image to clipboard (Ctrl-c) 64
Import .PNP phase noise data (N) 64
Import ASCII phase or frequency data (L) 64
Export ASCII phase data (x) 65
Export ASCII frequency data (X) 65
Export binary phase data 66
Export phase data to Stable32 (Ctrl-x) 66
Export xDEV trace 67
Export AM/PM noise trace 67
Export AM/PM spur table 67
Print image 68
Scale file dialogs by window size 68
Warn before exiting with unsaved plots 68
Reset all parameters, options, and settings at next startup 68
Close selected plot (Del) 68
Close all visible plots (Home) 69
Delete selected plot's .TIM file (Ctrl-Del) 69
Quit (q or Esc) 69

Edit Menu 71
Trace properties (e) 71
Flatten selected or zoomed phase data (Ctrl-f) 73
Remove selected or zoomed phase data (F4) 74
Subtract global linear phase trend (frequency offset) (Ctrl-o) 75
Subtract global linear frequency trend (drift line) (Ctrl-l) 75
Subtract quadratic linear frequency trend (drift curve) (Ctrl-q) 75
Undo last flatten or subtract operation (Ctrl-z) 75
Trace Menu 77
Phase/frequency traces begin at zero (z) 77
Show linear phase/frequency residual (r) 77
Show linear phase/frequency trend (Ctrl-t) 79
Phase/frequency Y axis unlocked in zoom mode (y) 80
Averaging window for phase/frequency traces (g) 80
Increase averaging window (Ctrl +) 80
Decrease averaging window (Ctrl -) 80
Draw xDEV traces with spline interpolation (i) 82
Show xDEV error bars (Ctrl-e) 82
Clip xDEV traces by noise bandwidth (Ctrl-b) 83
Clip xDEV traces by confidence (Ctrl-v) 83
Show correlation gain for selected noise trace (Ctrl-g) 83
Show FFT segment filter slopes (Ctrl-i) 83
Show imaginary part of cross spectrum (Ctrl-F3) 84
Show estimated instrument noise (F2) 84
Mark spurs in noise traces (Ctrl-m) 85
Suppress spurs in noise traces (Ctrl-s) 85
Smooth noise traces (Ctrl-w) 85
Show raw PN channel trace(s) (Ctrl-r) 85
Show AM noise in PN view (F8) 86
Tick marks (k) 86
Toggle trace thickness for current measurement (T) 87
Display Menu 89

Edit colors 89
High contrast (C) 89
Numeric table (Ctrl-n) 90
Show cursor time (Ss) 90
Show cursor time (Hh:Mm:Ss) 90
Show cursor time/datestamp 90
Do not show cursor values 90
Browse plots one at a time (b) 91
Overlay all loaded plots (o) 91
Toggle visibility of selected plot (v) 91
Select next plot in chart (+ or down arrow) 91
Select previous plot in chart (- or up arrow) 91
Move selected plot up (Ctrl-up arrow) 91
Move selected plot down (Ctrl-down arrow) 91
X zoom in ( ]) 92
X zoom out ([ ) 92
Y zoom in (}) 92
Y zoom out ({) 92
Decrease font size ((or Ctrl-mouse wheel) 92
Increase font size ( ) or Ctrl-mouse wheel) 92
Legend Menu 93
Measurement Menu 95
Masks Menu 97
Clear mask selection 97
User-defined mask entries 97
Edit mask definitions 98
Scripts Menu 99
Creating a new test script 99
Run script 100
Edit script 100
Delete script … 100
Run last-executed script (Shift-Space) 100

Toggle script console for selected plot (F11) 101
Stop all running scripts (F12) 101
Acquire Menu 103
Miles Design TimePod 103
Acquire from counter in Talk-Only mode 103
Acquire from live ASCII file 104
HP 53131A/53132A/53181A 104
HP 53220A/53230A 104
HP 5335A 104
HP 5370A/B 104
HP 5371A/5372A 104
Philips/Fluke PM6680 104
Picotest/Array U6200A series 104
Stanford Research SR620 104
Wavecrest DTS-2050/2070 series 104
Symmetricom 5115A / 5120A / 5125A (Frequency stability) 105
Symmetricom 5115A / 5120A / 5125A (Phase noise) 105
Stop/repeat acquisition (Space) 105
Abort and retrigger selected acquisition (Ctrl-a) 106
Keep and retrigger selected acquisition (Ctrl-k) 106
Enable deferred acquisition (Ctrl-d) 107
Trigger deferred acquisition(s) (Enter) 107
Configure deferred acquisition 107
Help Menu 109
User guide (F1) 109
About TimeLab 109
Debug mode 109
Check for updates (Ctrl-u) 109
Appendix: Some examples of residual performance 111
Appendix: JavaScript API Function Reference 115
Appendix: The STREAM.EXE Phase/Frequency Data Server 135
Launching STREAM.EXE 136

Using STREAM.EXE 137
/serial:<sernum> 137
/file:<filename> 137
/logfile:<filename> 138
/port:<portnum> 138
/msglvl:<0-5> 139
/format:<P, F, TSC> 139
/window:<samples> 140
/rate:<1, 10, 100, 1000> 141
/timestamp:<s, MJD> 143
/sep:<character> 143
/ref:<Hz> 144
/input:<Hz> 144
/autoupdate 145
/noexit 146
/notcp 146
/nowarnings 146
Appendix: Schematic diagrams and service notes 147
Appendix: Legal and regulatory notices 155
Federal Communications Commission Statement 156
EC Declaration of Conformity 156
Performance Certification and Validation 157
Limited Warranty 157
Software License: TimeLab 157
Software License: FFTSS 158
Software License: FIDLIB 158
Software License: V8 JavaScript Engine 163



Introduction

The **TimePod® 5330A Programmable Cross Spectrum Analyzer** measures the amplitude, phase and frequency stability of RF sources and two-port devices at frequencies from 0.5 MHz to 30 MHz. Results can be viewed at timescales ranging from femtoseconds to days.

Measurements made by the TimePod 5330A include the following:

- Real-time 'strip charts' of phase and frequency differences at subpicosecond precision
- Absolute frequency counts at 13+ digits per second, 17 digits maximum
- Allan deviation (ADEV) typically less than 1E-13 at t=1s
- Modified Allan deviation (MDEV), Hadamard deviation (HDEV), and time deviation (TDEV)
- Phase noise and AM noise at offsets from 0.01 Hz to 100 kHz and levels below -170 dBc/Hz
- RMS-integrated time jitter with less than 100 fs residual jitter from 0.01 Hz to 100 kHz
- RMS-integrated phase noise, residual FM, and SSB carrier/noise ratio

Using high performance host-based DSP techniques on a Windows® PC, all of these measurements can be made simultaneously. Real-time results appear as you watch – and you can save, view, compare, export, or print them at any time. Accuracy and stability are inherited from a user-supplied external reference which can run at any frequency within the supported range, with no calibration required by the instrument itself.

Specifications

Input frequency and level	0.5 MHz – 30 MHz, -5 dBm - +20 dBm, 50 ohm TNC-F
Reference frequency and level	0.5 MHz – 30 MHz, -5 dBm - +20 dBm, 50 ohm TNC-F
Input/reference VSWR (0.5-25 MHz)	1.5:1 or better
Input/reference port isolation (10 MHz)	130 dB or better
Maximum allowed DC at any RF input	+/- 5V
Allan deviation (5 MHz-25 MHz, t=1s)	1E-13 minimum, 5E-14 typical (50 Hz ENBW)
Allan deviation (5 MHz-25 MHz, t=1000s)	5E-15 minimum, 1E-15 typical
Phase stability (5 MHz)	Less than 10 ps/hour after 2 hour warmup Typically below 3 ps/hour
Residual phase noise floor (5 MHz, 1 Hz)	-140 dBc/Hz minimum, < -145 dBc/Hz typical
Residual phase noise floor (25 MHz, 1 Hz)	-130 dBc/Hz minimum, < -135 dBc/Hz typical
Residual phase noise floor (5 MHz, 10 kHz)	-170 dBc/Hz minimum, < -175 dBc/Hz typical
Residual phase noise floor (25 MHz, 10 kHz)	-165 dBc/Hz minimum, < -170 dBc/Hz typical
Residual AM noise floor (5 MHz, 10 kHz)	-160 dBc/Hz minimum, < -165 dBc/Hz typical
Spurious responses (5 MHz, 1 Hz-100 kHz)	Less than -100 dBc (phase noise) or -90 dBc (AM noise) Typically below -120 dBc
Physical dimensions	280 mm x 120 mm x 75 mm, 1 kg 11” x 5” x 3”, 2 pounds
Power requirements	90-264 VAC, 47-63 Hz, < 25W 3-pole AC inlet IEC320-C14
Ambient temperature	15C to 35C operating, -20C to +50C storage 60F to 95F operating, 0F to 125F storage

Note: Due to the use of cross correlation to cancel instrument noise, the phase noise and AM noise floors depend strongly on the measurement time and available signal levels. Residual specifications assume that measurements are made after a two-hour warmup period in a temperature-stable environment with +15 dBm at both INPUT and REF IN jacks and **Trace→Smooth noise traces** enabled. Under these conditions, 5-10 MHz signals are typically measurable to better than -170 dBc/Hz at offsets > 10 kHz after less than 10 minutes. Quieter signals, lower-amplitude signals, signals near the frequency-coverage limits, and measurements of very low close-in noise may require more time to converge.

Getting started

What's in the box?

Please check the contents of your package carefully upon arrival. Each TimePod 5330A unit should be accompanied by the following items:

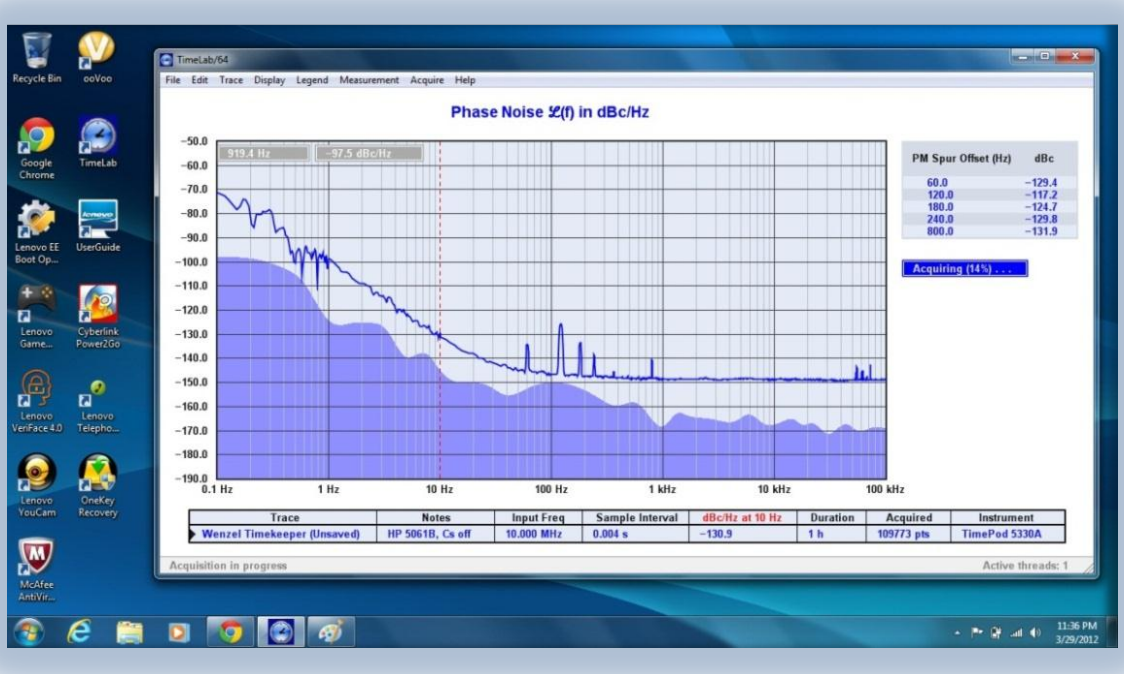
- (1) USB 2.0 cable, A Male / B Male
- (1) Power supply
- (2) TNC-M / BNC-F coax adapters
- (2) 1" (25.4 mm) SMA-M / SMA-M coax jumpers (preinstalled)

Additionally, each 5330A delivered to North American customers includes a standard IEC320 / NEMA 5-15P power cord for 120V service. For operation in other countries, the 5330A's power supply accepts 50/60 Hz AC power at all standard line voltages from 100V to 240V, and is compatible with IEC320-C13 power cords available locally.

Sales and technical support

For prompt assistance, contact Miles Design at the email address or telephone number on the inside front cover of this manual. When leaving a message, please include your name, organization, callback number, and preferred time to call (if any). Telephone calls and email are typically returned within 12-24 hours.

USB driver and software installation


TimeLab/64
File Edit Trace Display Legend Measurement Acquire Help
Phase Noise ℒ(f) in dBc/Hz
PM Spur Offset (Hz) dBc
60.0 -129.4
120.0 -117.2
180.0 -124.7
240.0 -129.8
800.0 -131.9
Acquiring (14%) . . .
0.1 Hz 1 Hz 10 Hz 100 Hz 1 kHz 10 kHz 100 kHz
Trace Notes Input Freq Sample Interval dBc/Hz at 10 Hz Duration Acquired Instrument
Wenzel Timekeeper (Unsaved) HP 5061B, Cs off 10.000 MHz 0.004 s -130.9 1 h 109773 pts TimePod 5330A
Acquisition in progress
Active threads: 1

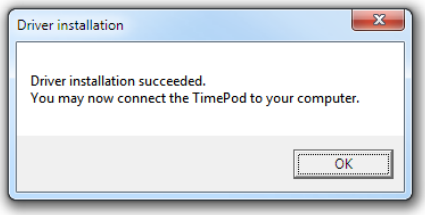
The current TimeLab release may be downloaded from http://www.miles.io/timelab/readme.htm.
Please install the most recent version of TimeLab before using your new 5330A!

The latest edition of this manual is available at http://www.miles.io/TimePod_5330A_user_manual.pdf.

TimeLab runs on Intel® or AMD® x86-based PCs equipped with Microsoft Windows® XP SP2 or later versions. It supports a wide variety of time and frequency measurement instrumentation in addition to Miles Design's own instruments. Minimum system requirements are 100 MB of disk space, 1 GB RAM and a CPU with SSE2 support. A dual- or quad-core processor is strongly recommended.

By default, TimeLab will automatically check the Miles Design web site on a weekly basis and inform you if a newer version is available for download. Updates are always free of charge. To configure or disable automatic update notifications, select **Help→Check for updates**.

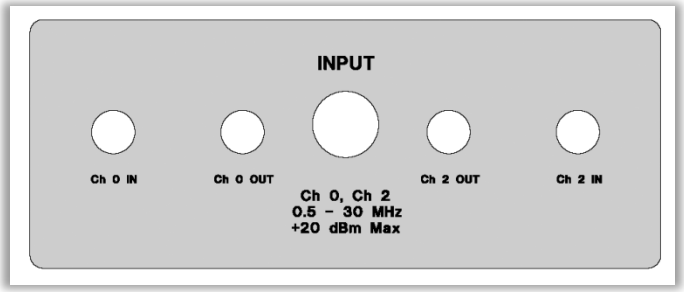
For best results, TimeLab should be installed prior to connecting the 5330A to your PC's USB port for the first time. Before exiting from the install program, make sure that the box labeled **Install TimePod 5330A USB Driver** is checked. This will help Windows locate the USB driver without further intervention when the 5330A is connected. After the driver has been successfully installed on your system, a confirmation message will appear.



To perform measurements with the 5330A, an Intel Core 2 Duo or faster processor is required. Referring to the benchmarks at http://www.cpubenchmark.net/common_cpus.html, the minimum PassMark score for reliable acquisition falls in the 1,600 to 2,000 range.

Use of a system with inadequate CPU performance may result in acquisition errors, often accompanied by a flashing red fault indication on the 5330A's status LED. It may be necessary to disable one or more measurement types to achieve reliable operation in such cases. For example, if you are interested only in phase/frequency stability, uncheck the **Phase Noise** and **AM Noise** boxes in the **Available Measurements** area of the TimePod acquisition dialog.

For improved performance, a 64-bit version of TimeLab will automatically be installed on x64-based systems. If you need to record long phase records with the 5330A (or any other equipment), an x64 system with several GB of RAM is recommended.



Front panel features

Four SMA jacks and one TNC jack are present on the 5330A's input jack panel. In most applications, the input signal from the device under test should be applied to the centrally-located TNC jack labeled **INPUT**.

Internally, the **INPUT** jack is connected to a 50-ohm 0° RF splitter whose two outputs are routed to the SMA jacks labeled **Ch 0 OUT** and **Ch 2 OUT**. These jacks are normally connected to the adjacent **Ch 0 IN** and **Ch 2 IN** jacks using two SMA jumpers.

Although TNC interconnects offer superior mechanical stability, BNC-TNC adapters are also provided for convenience. Regardless of your choice of coax fittings, double-shielded cables such as RG223 or RG400 are recommended for low-level measurements that may be affected by crosstalk and environmental interference. Use of RG58 and other single-shielded cables can cause artifacts in stability and noise plots.

Input signals should be greater than 0 dBm for best performance -- +5 to +15 dBm is recommended. The 5330A's specifications assume that sine-wave signals from 50 ohm sources are applied to the **INPUT** and **REF IN** jacks, but you can also measure CMOS and other square-wave clocks with the help of a simple L-network (resistive or otherwise) to attenuate the signal and increase the load impedance where necessary.[1]

Spur performance may be compromised with non-sinusoidal inputs. This is especially true at lower frequencies where multiple harmonics fall within the 0.5 – 30 MHz passband.

Why the added complexity? Wouldn't a single input jack be enough?

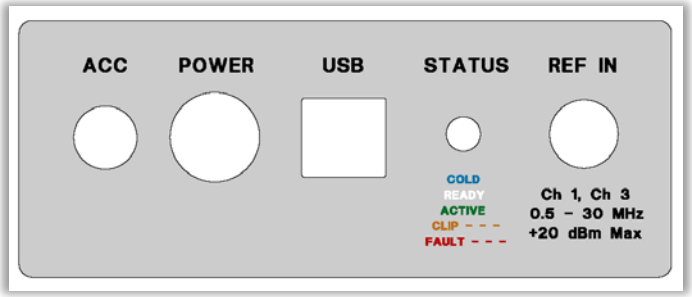
The 5330A is really two identical instruments in one box. Each of these two "instruments" consists of a pair of software-defined HF receivers for the DUT and reference input signals. All four receiver channels are implemented with high performance 16-bit RF ADCs. During cross-correlated phase noise and AM noise measurements, the even-numbered channels (**0** and **2**) are assigned to the input ADCs, while odd-numbered channels (**1** and **3**) are associated with the reference ADCs.

Because the instrument noise originating within each ADC tends to be uncorrelated with the noise from that ADC's counterpart in the other "instrument," the differential-mode noise falls out of the cross-spectrum average, approaching zero amplitude over time. Ideally, only the desired phase and amplitude information will remain.

Like the input-channel ADCs (0 and 2), ADC channels 1 and 3 are fed identical copies of the signal at the **REF IN** jack through an RF splitter, but this signal path is only routed internally. Making the individual *input* ADC channels available with the **Ch 0 IN** and **Ch 2 IN** jacks is worthwhile, though, because it opens up some possibilities for advanced measurements.

For example, you can make low-noise measurements at VHF and microwave frequencies far beyond the 5330A's rated 30 MHz limit by using two identical downconverters to mix the DUT signal down to an IF in the supported 0.5 – 30 MHz range. If the Ch 0 and Ch 2 downconverter local oscillators are uncorrelated, their phase noise and AM noise contributions will disappear from the cross spectrum average over time, just like the ADC noise!

[1] Both the **INPUT** and **REF IN** jacks are coupled to their respective input splitters via 0.1 uF 100V capacitors. These capacitors are also present at the **Ch 0 IN** and **Ch 2 IN** jacks. However, application of DC levels greater than 5V peak to any RF signal input is not recommended.



USB 2.0 High Speed support is required for all 5330A acquisitions. In most cases the use of passive or active USB hubs with the 5330A is acceptable, but if connection problems occur, you may find it helpful to connect the supplied USB cable directly to the host PC.

In general, measurement errors are *much* more likely to be caused by insufficient CPU resources than by USB connectivity problems.

Rear panel features

The rear jack panel provides a TNC-F input for the external reference signal required in all 5330A measurements. All guidelines regarding signal levels and interconnect choices for the **INPUT** jack also apply to the **REF IN** input. Suggestions for choosing appropriate reference sources appear below.

Additionally, the 5330A's rear panel provides a 5-pin DIN jack for connection to the power supply (**POWER**), an 8-pin Mini DIN jack for future accessory expansion (**ACC**), and a USB Series 'B' receptacle for PC connectivity (**USB**).

The **STATUS** indicator is a high-brightness RGB LED that reveals the instrument status at a glance:

Blue Present for the first few minutes while the 5330A's internal OCXO warms up.

White The 5330A is ready to acquire data. Stability measurements at 50 Hz and higher bandwidths may be performed as soon as the LED turns from blue to white, although a 30-minute warmup period is recommended. Input/reference drift warnings may occur with shorter warmup periods.

Green A measurement is either pending or in progress. No fatal errors have occurred.

Flashing purple Excessive signal level is present at either **INPUT** or **REF IN**, causing input clipping in one or more ADC channels. An unexpected increase in signal level after measurement has begun can cause this indication. This is not a fatal error, but acquired data is questionable. Damage may occur at input levels greater than +20 dBm.

Flashing red A USB data overrun has occurred. This is a fatal error. Insufficient CPU power (or disconnection of the USB cable during a measurement) is typically responsible. You may be able to run the measurement successfully if you uncheck one or more of the **Available Measurements** options in the acquisition dialog.

Choosing an external reference

In addition to the host PC and an appropriate power source, all measurements made with the 5330A require an external reference to be supplied at the REF IN jack. Virtually all aspects of measurement performance – accuracy, repeatability, noise floor, spurious responses – depend on your ability to provide the best reference signal possible.

But what does the "best reference possible" mean? It depends on the goal(s) of your measurement. Few reference sources are ideal for both short- and long-term measurements. Typical "house clocks" that distribute 5 or 10 MHz signals at levels between +5 and +20 dBm often work well for ADEV and other time/frequency measurements, but atomic and GPS standards that are often used as sources for centralized clock distribution may exhibit more short-term phase noise than an undisciplined crystal oscillator. Distribution amplifiers can also add substantial noise of their own, as well as phase drift at longer timescales due to temperature sensitivity. Finally, even when using the best standards, distribution amps and double-shielded cables, it's almost impossible to build a large clock distribution network that's free of environmental spurs. As a result, you should consider using a standalone low-noise OCXO for phase noise and AM noise measurements.

Unlike most other stability and noise analyzers, the 5330A can work with any reference whose frequency lies within its specified range (0.5-30 MHz), regardless of the input frequency from the device under test (DUT). **Frequency readings and phase noise levels are always referred to the frequency at the INPUT jack.** For example, if your 10 MHz DUT's phase noise when measured with a 5 MHz reference is -160 dBc/Hz at 10 kHz and its Allan deviation is 3E-12 at t=10s, you will still measure -160 dBc/Hz and ADEV=3E-12 if you switch to a 10 MHz reference, or one at any other frequency.

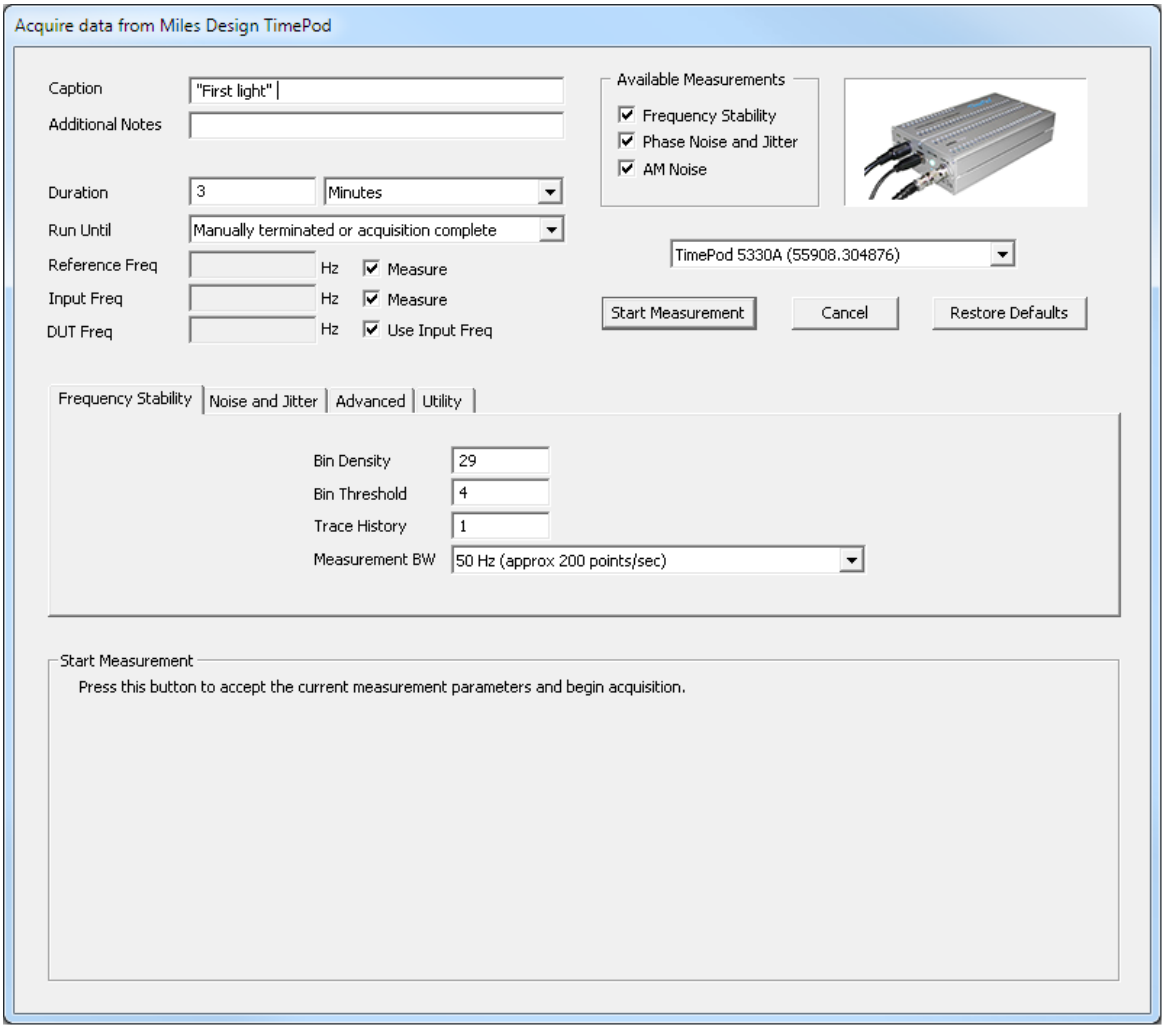
The only complication is the need to handicap the reference's phase noise and FM/PM spur amplitudes by $20*\log10(F_{DUT}/F_{REF})$ dB when the DUT and reference frequencies differ. This is the same consideration that applies at all other times when a signal undergoes broadband frequency multiplication or division. The effective phase noise and spur performance of the reference source would be 6 dB worse in the example above where a 5 MHz reference is used to characterize a 10 MHz device. Likewise, the reference's long-term stability characteristics (e.g., Allan deviation) would be degraded by a factor of 2.

Ideally, the phase noise of your reference after any applicable $20*\log10(F_{DUT}/F_{REF})$ correction should be at least 8-10 dB lower than the expected performance of the DUT at the offsets of interest. The reference's phase noise will not affect the measurement to any great extent as long as this margin is maintained.[2]

[2] This being said, it's sometimes useful to measure the phase noise or stability of a DUT by using an identical device as the reference. In such a case, where the phase noise of the reference and DUT is presumed to be identical but uncorrelated, the resulting PN graph will be 3 dB too high, while Allan deviation and related measurements will be artificially elevated by a factor of sqrt(2). You can use the **Rescale Phase** field in TimeLab's **Edit→Trace properties** dialog to correct the ADEV of two identical devices by scaling the phase data by 0.707. Similarly, the phase noise may be corrected by entering -3 in the **PN Gain** field to lower the trace by 3 dB.

Making your first measurements

To paraphrase Kernighan and Ritchie[3], TimeLab is not a large program, and it's not well served by a large manual. Likewise, the TimePod 5330A hardware is easy to work with after some basic measurement concepts are understood. As an introductory exercise – or a quick operational check – you may wish to perform a measurement using the default acquisition parameters in TimeLab.



To get started, simply connect the 5330A's USB cable to your PC and launch TimeLab. Once TimeLab is running, select the **Acquire→Miles Design TimePod** menu option. The dialog box that pops up should contain a list of available TimePod device(s) in a dropdown box at upper right, just below the instrument photo.

[3] *The C Programming Language*, Second Edition (1988), Bell Telephone Laboratories

Assuming the appropriate TimePod 5330A instrument is selected, the acquisition parameters are all valid, you've connected stable reference and DUT input sources to the 5330A, and you've allowed at least a few minutes' warmup time, all you need to do is press the **Start Measurement** button. The acquisition dialog will disappear and the **STATUS** indicator on the 5330A should change from white to green. Over the next ten seconds or so, various informational messages will appear in the TimeLab status bar at the bottom of the program's main window as the software characterizes the applied input and reference signals.

If all is well, the status bar message should soon change to "Acquisition in progress." Measurement data will now begin to appear in graphical form. Unless you've selected a different view from the **Measurement** menu, you will see an Allan deviation (ADEV) plot for your DUT, rendered in real time at 100 points per second.

Unless you changed the **Duration** value, the measurement will run for three minutes. This will be long enough to allow the ADEV trace to reach τ=one minute, and for the phase noise trace to reach 1 Hz. You can try any of these suggested actions at any time, either during the measurement or after it finishes:

- Left-click on the ADEV graph or any other log-log plot. Observe that the red *spot cursor* can be placed at any vertical column. The graph's Y value at the spot cursor will be displayed in the *legend table* beneath the graph area, updated in real time to reflect the latest reading. At the same time, ADEV values at assorted tau periods should be displayed in a chart on the right side of the graph.

- Left-click and drag a box anywhere on the plot. When you release the left mouse button, the view should zoom in to magnify the specified area. You can drag an edge (or a corner) of the box slightly outside of the graph area to expand the plot in that direction. Right-click to return to the unzoomed view.

- Try selecting a few of the different graph types in the **Measurement** menu. Assuming you didn't uncheck any of the **Available Measurements** options in the acquisition dialog, all of the different graph types should be visible, updating continuously in a "live" fashion as long as the measurement is still in progress.

- Now is a good time to start picking up the keyboard shortcuts. Instead of selecting **Measurement→Frequency difference**, try the **f** key. To return to the Allan deviation plot, use **a**. Uppercase **A** selects AM noise, while uppercase **P** selects phase noise. (In some cases, the TimeLab features and commands described in this manual will be referenced by their hotkeys rather than their menu entry names, after the menu entries have been introduced.)

- Save the measurement data to a .TIM file with **File→Save image or .TIM file (s)**. A .TIM file is an ASCII text file that contains the data needed to recreate all of the available graphs

associated with a given measurement. This data includes the entire phase record, as well as the FFT bins from the phase noise and AM noise plots. You don't have to wait for the measurement to end; at any time during a measurement, you can save the available data to a .TIM file or load another .TIM file for display alongside the existing plot(s) that have been loaded or acquired.

- Use **File→Load .TIM file** to read the .TIM file you just saved. Now you should have two copies of the same plot in memory. If the measurement is still in progress, you can switch to the **p)hase** or **f)requency difference** views to watch its phase record continue to grow while the saved copy remains unaffected.

For "power users," three of the most common commands in TimeLab are **Display→Browse plots one at a time (b)**, **Display→Overlay all loaded plots (o)**, and **Display→Toggle visibility of selected plot (v)**. Like many other commands, the **b**, **o**, and **v** commands operate on the so-called "selected" plot. If you've followed along with the suggested command demonstrations above, then you should have two (or more) plots loaded, and you can experiment with these concepts.

- First, select any desired **Measurement** view and press the **b** key to enter Browse mode. Observe that only one of your plots is now visible.

- Furthermore, notice that two things happen when you press the up/down arrow keys (**Display→Select next plot in chart** / **Display→Select previous plot in chart**). A small black triangular cursor at the leftmost edge of the legend table moves up and down… and the plot that it points to is also the one that becomes visible. In **Display→Browse** mode, entries in the legend table corresponding to all of the unselected plot(s) are grayed out, and those plots aren't shown in the graph.

- You can put things back to normal by pressing **o** to return to **Display→Overlay** mode. The idea of a "selected" plot still exists, but it no longer determines which plots are visible in the legend table and graph areas.

- In **Display→Overlay** mode, all of the loaded plots are visible, all of the time, unless individually hidden with the **Display→Toggle visibility of selected plot (v)** feature. To avoid confusion the **v** key doesn't do anything in **Display→Browse** mode, but in **Display→Overlay** mode it does just what it sounds like – it toggles the visibility of the selected plot in both the graph area and the legend table. This panoply of options may seem confusing at first, but it's nothing compared to the jumble of line segments and shaded areas that can appear on the graph when all nine available slots in the legend table are populated with visible plots!

- Regardless of the choice of **Display→Browse** or **Display→Overlay** mode or the visibility status of any given plot, the "selected plot" concept is still useful for indicating which plot should contribute to the frequency-count chart in the **Measurement→Frequency**

Difference (f) view, the sigma-tau charts in the various xDEV measurement views, and the spur tables in the noise measurement views. The selected plot is also the only one that's saved, exported, modified, edited, or moved up and down in the legend table when the corresponding commands are issued.

The last point above is an important one – many users new to TimeLab are surprised to discover that a single .TIM file doesn't represent an entire screen full of plots. Each plot must be saved, loaded, and otherwise processed individually. None of the commands on the File or Edit menus in TimeLab operate on more than one plot at a time, except for **File→Close all plots (Home)** and the various image-based operations like **File→Print image (Ctrl-p)**, **File→Copy image to clipboard (Ctrl-c)**, and **File→Save image or .TIM file (s)** when the latter command is used to save a .png, .bmp, .tga, .gif, or .pcx image of the entire screen.

Tips for new users

- In TimeLab, **almost everything you can do from the menus has a keyboard equivalent**, in many cases a single key. Time spent becoming familiar with the keyboard shortcuts will be rewarded!

- The acquisition dialog (**Acquire→Miles Design TimePod**) contains a **Utility** tab with a button labeled **Update Firmware** which will upload new .FX2 or .BIT files to nonvolatile memory on the 5330A. .FX2 files contain 8051 firmware for the onboard USB 2.0 controller, while .BIT files contain FPGA configuration data.

  TimeLab v1.003 (RC4) contains new versions of both files, **MD5330A_100.bit** and **MD5330A_100.fx2**. These files are present in the directory where TimeLab was installed – normally c:\Program Files\Miles Design\TimeLab. **Users of TimePod 5330As prior to S/N 55985 should upgrade their hardware with both of these files, particularly if USB connectivity problems have been encountered.**

- Use the 5330A in a room with good ambient temperature control for best results. Avoid placing the hardware near HVAC vents or other sources of drafts when making high-performance measurements.

- Also for best results, avoid placing the 5330A immediately next to equipment that may operate at high temperatures, as well as equipment that should not be exposed to heat. Normal case temperatures may reach 40C to 45C after several hours' operation.

- Do not disturb the measurement setup mechanically, move cables around, or otherwise interact with the environment unnecessarily during acquisition. After the measurement ends, you can use TimeLab's phase-record editing features to get rid

of known glitches, but nothing can be done to restore a corrupted noise plot. Any glitches that affect the phase noise or AM noise plots will require a restart.

- Every measurement in TimeLab runs in its own background thread, so you can load, save, and manipulate plots or even launch additional acquisitions with other instruments while an acquisition is running. This is subject to available memory and CPU resources, of course. A typical TimePod 5330A acquisition takes about 16 MB/sec of USB bandwidth and about one core's worth of CPU horsepower.

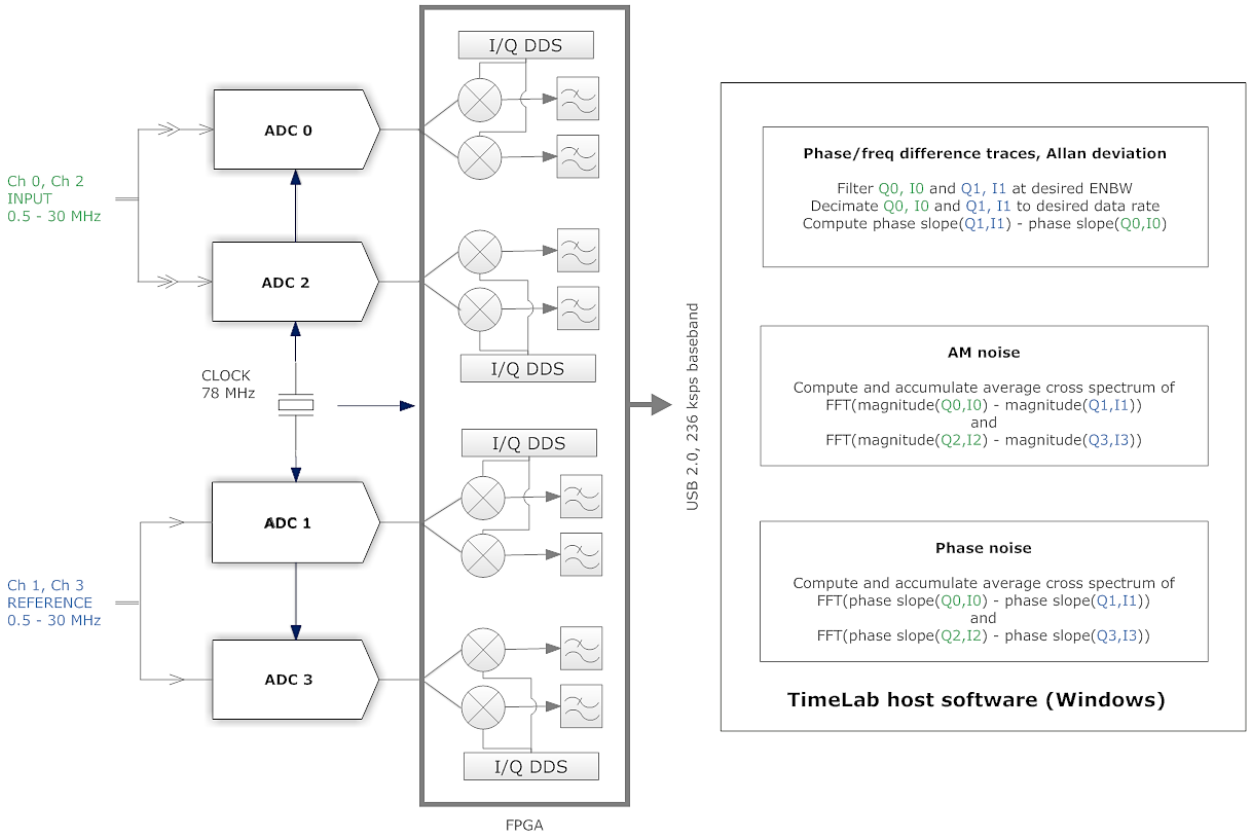
- When equipped with multiple USB host controllers, the fastest available PCs can acquire data from up to four TimePods concurrently. Most PCs will be limited to two concurrent acquisitions at most.

- The acquisition dialogs in TimeLab all have large "mouseover" help windows. The help text serves as hardware-specific documentation for the various analyzers and counters supported by TimeLab. Read it carefully before changing any fields from their default values.

- Attempting to run **Frequency Stability** measurements with large phase records and high data rates may result in "Couldn't allocate phase record" or similar messages indicating a lack of available RAM. Running on an x64 system can help avoid memory problems, as they can access much more RAM than legacy 32-bit Windows systems.

- When contacting Miles Design for technical support, it's a good idea to attach .TIM file(s) associated with the issue, rather than image files. .TIM files greater than 3 MB in size should be zipped or otherwise compressed. *If you don't receive a reply within 48 hours, please email again (without the attachment) to confirm delivery.*

A brief architectural note



It can be helpful to understand some basic details about the TimePod 5330A's DSP topology, especially when measurements may be constrained by available memory and CPU resources. A key observation is that each of the three basic measurement types **-- Frequency Stability**, **Phase Noise and Jitter**, and **AM Noise** -- uses a completely separate data record and internal pipeline.

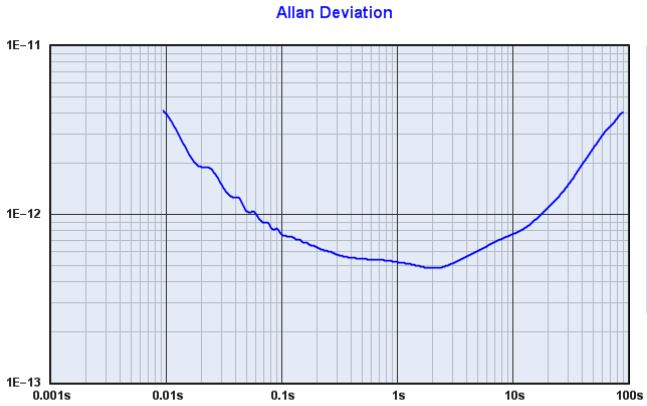
The "phase record" mentioned in the documentation and help text has nothing to do with **Phase Noise** measurements, for example. Instead, the phase record and its associated controls are used only for **Frequency Stability** measurements. These include ADEV, MDEV, TDEV, and HDEV, as well as the frequency- and phase-difference plots.

Referring to the diagram above, the phase record is created from two parallel streams of 236 ks/sec complex data samples arriving from one input channel (Q0,I0) and one reference channel (Q1,I1). These two streams are decimated to accommodate the requested measurement bandwidth (ENBW) – typically 5 to 500 Hz. The phase differences between corresponding samples in the two low-bandwidth streams are then calculated and used to construct the various xDEV plots as well as the frequency-difference plot.

So, when you save a **Frequency Stability** measurement to a .TIM file, you're actually saving these low-bandwidth phase differences. If you use **Edit→Remove selected or zoomed phase data (F4)** or the various flattening or trend-removal functions to edit parts of the phase record after acquisition, the

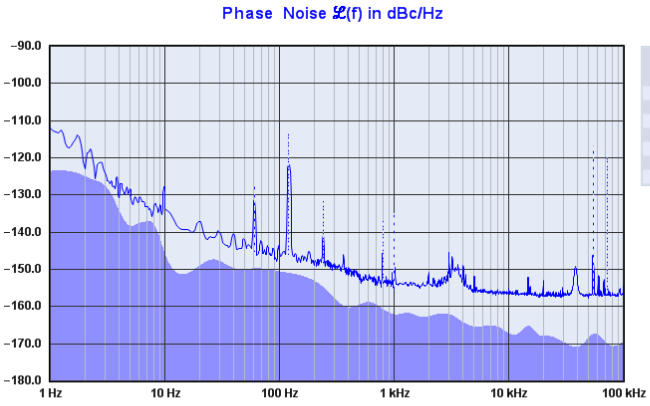
xDEV and frequency-difference plots will be recalculated, but *nothing will happen to the phase noise or AM noise plots*. These graphs are created by calculating phase and/or amplitude differences for all four baseband streams -- Q0,I0, Q1,I1, Q2,I2, and Q3,I3 – at the full 236 ks/sec data rate. The resulting data is then processed by FFT and cross-spectrum averaging routines. Because it would be impractical to retain the original high-bandwidth complex baseband data or any phase or amplitude information derived from it, only the averaged FFT output bins are written to the .TIM file when saving the phase noise or AM noise portion of a measurement.

Making measurements



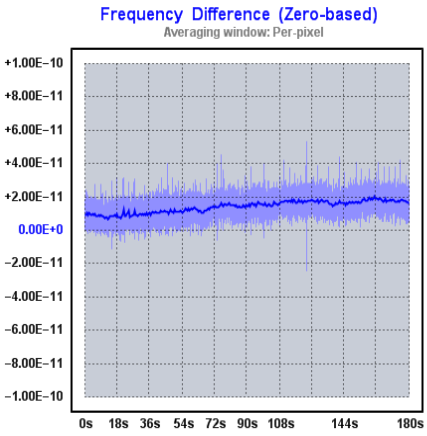
Tau	Sigma(Tau)
0.01s	3.87E−12
0.02s	1.94E−12
0.04s	1.26E−12
0.08s	8.43E−13
0.1s	7.63E−13
0.2s	6.42E−13
0.4s	5.58E−13
0.8s	5.37E−13
1s	5.26E−13
2s	4.85E−13
4s	5.69E−13
8s	7.24E−13
10s	7.73E−13
20s	1.11E−12
40s	2.02E−12
80s	3.78E−12

Trace	Input Freq	Input Amplitude	Ref Freq	Sample Interval	ENBW	Elapsed	Instrument
HP 10811-60109	10.000 MHz	13 dBm	10.000 MHz	0.004 s	52.7 Hz	3m 0s	TimePod 5330A



PM Spur Offset (Hz)	dBc
60.0	−126.6
120.0	−113.2
240.0	−131.7
800.0	−137.0
1000.0	−135.1
54910.0	−116.8
72810.0	−119.6

Trace	Input Freq	Input Amplitude	Ref Freq	Sample Interval	ENBW	Elapsed	Instrument
HP 10811-60109	10.000 MHz	13 dBm	10.000 MHz	0.004 s	52.7 Hz	3m 0s	TimePod 5330A



Origin	Drift (Hz/sec)	Drift (Hz/min)
+8.64E−12	+6.14E−7	+3.68E−5

Avg Time (s)	Freq (Hz) at 180s	Error
0.100	10 000 000.013 018 839	+1.30E−9
0.300	10 000 000.013 017 127	+1.30E−9
1	10 000 000.013 019 862	+1.30E−9
3	10 000 000.013 023 891	+1.30E−9
10	10 000 000.013 027 977	+1.30E−9
30	10 000 000.013 032 110	+1.30E−9
100	10 000 000.013 020 335	+1.30E−9

Trace	Input Freq	Input Amplitude	Ref Freq	Sample Interval	ENBW	Elapsed	Instrument
HP 10811-60109	10.000 MHz	13 dBm	10.000 MHz	0.004 s	52.7 Hz	3m 0s	TimePod 5330A

Three views of the same TimePod 5330A measurement, in which a GPS-disciplined oscillator was used as a reference to measure a high-quality HP 10811 OCXO.

The Allan Deviation plot can reveal frequency stability at timescales from milliseconds to days, while the Phase Noise plot can show you the signal's USB spectral signature at offsets from 0.01 Hz to 100 kHz. (Here, the GPS clock dominates the noise and spur picture by a wide margin!)

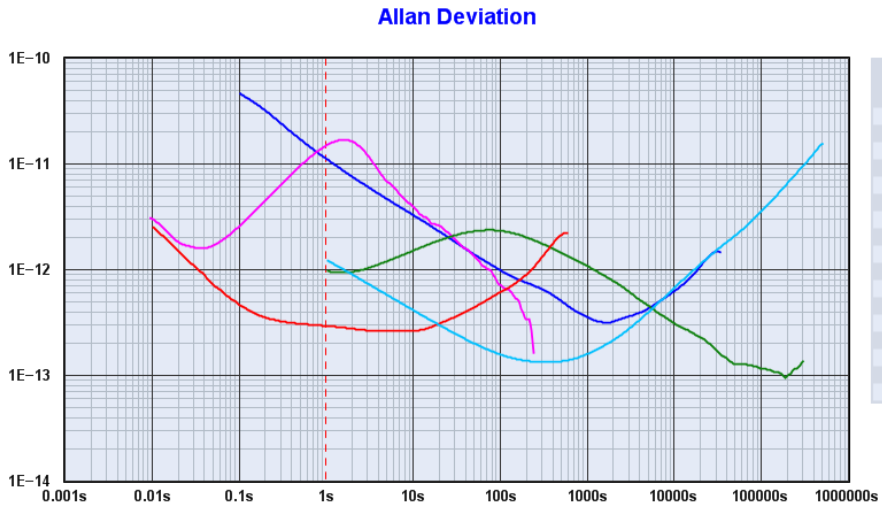
Valuable insights can be gained by looking at your data from more than one point of view. For instance, the visible ADEV ripple below 0.1 second corresponds to the 60-Hz and 120-Hz AC line spurs in the phase noise plot.

Meanwhile, drift and sporadic instability can be examined in the Frequency Difference display. When using a 5330A, the frequency-count chart offers over 13 digits per second of *usable* precision at "gate times" from 0.1 second to over 1000 seconds.

"What's all this ADEV stuff, anyhow?"

With apologies to the late Bob Pease and his editors at *Electronic Design*, here are a few tips to help you understand what Allan deviation (ADEV) graphs really tell you, and how to get the most out of the statistical deviation measurements made by TimeLab.

Allan deviation is defined as the *two-sample deviation of fractional frequency differences at a given time interval*. While technically correct, such definitions may be unenlightening to new users. Often, the first question asked is something like, "Why did my plot stop at 1000 seconds? It took an hour to run!"



Tau	Sigma(Tau)
0.1s	4.74E−11
0.2s	3.22E−11
0.4s	2.01E−11
0.8s	1.29E−11
1s	1.13E−11
2s	7.63E−12
4s	5.27E−12
8s	3.70E−12
10s	3.31E−12
20s	2.31E−12
40s	1.59E−12
80s	1.11E−12
100s	9.96E−13
200s	7.62E−13
400s	5.79E−13
800s	3.95E−13
1000s	3.58E−13

Trace	Notes	Input Freq	Sample Interval	ADEV at 1s	Duration	Instrument
LPRO-101 Rb	HP 5065A Rb	10 MHz	0.1 s	1.13E−11	20 h	TimePod 5330A
SRS PRS10 Rb	HP 10811-60109 in 5061A	10 MHz	0.004 s	1.53E−11	8 m	TimePod 5330A
HP 5061A Cs	Active maser	5 MHz	1.0 s	9.82E−13	7 d	TSC 5110A
Wenzel Timekeeper	Oscilloquartz BVA 8607/008	10 MHz	0.01 s	2.99E−13	20 m	TSC 5120A
Oscilloquartz BVA 8607/008	HP 5065A Rb	5 MHz	1.0 s	1.25E−12	12 d	TSC 5110A

The answer is that these plots are not simply linear depictions of frequency drift over time. (In TimeLab, that's the purpose of the **Measurement→Frequency Difference (f)** view.) Nor are they depictions of a single Allan deviation measurement that describes the clock's stability from one interval of a given duration to the next. Instead, ADEV and similar statistical graphs usually portray an entire collection of deviation measurements, each based on a list of phase or frequency data points spaced at an interval unique to that measurement.

Imagine that you own a watch that exhibits a certain accuracy in parts per million from one minute to the next. To arrive at this figure, you've checked your watch against a better clock several times, waiting one minute between each trial, and then calculated the Allan deviation of the resulting list of phase errors. (You might have used conventional standard deviation, but in this case you've heard, correctly, that STDEV has major drawbacks for timekeeping work.)

You're fairly certain you'll get a different result if you calculate the Allan deviation for a set of readings taken one week apart, and still another result for a series of readings taken annually. So you record at least a few data points at these intervals as well.

Now you'd like to draw a graph that shows off your watch's performance, perhaps for bragging rights at the neighborhood pub. What sort of plot would be needed to portray everything you've learned about the watch? The X axis obviously needs to be logarithmic, given that your timescales of interest range from minutes to years. The deviation figures will need to be plotted on a log axis as well, because they might be thousands of times worse at one-year intervals than they are at one-minute intervals. Finally, even though you recorded several years' worth of annual readings, your graph will need to end at the one-year point. There's simply not enough data to describe the watch's behavior at longer timescales.

If you keep these guidelines in mind as you graph your data, you'll end up with a legitimate ADEV plot. It will have three data points, each containing the result (*sigma*) of an independent calculation based on at least a few samples of the clock's frequency that were taken at a specific interval (*tau*). Lines or curves can be fitted to this collection of discrete points, and the result will be a plot like those drawn by TimeLab and other ADEV applications.

TimeLab's statistical capabilities aren't limited to Allan deviation -- the **Measurement** menu allows you to view any of four types of deviation plots, during or after acquisition. Formulas exist[4] to calculate Allan deviation and other statistics on the basis of both phase and frequency samples. TimeLab always works with phase data internally, using incremental calculations for fast processing and overlapped algorithms for high statistical quality. As with other measurements made by TimeLab, the statistical plots are updated simultaneously in real time.

Allan Deviation (a)

Allan deviation, or ADEV, is the square root of the Allan variance, $\sigma^2_y(\tau)$. Like its related statistics MDEV and HDEV, ADEV describes the fractional frequency deviation (σ, or *sigma*) of a set of samples taken at an interval τ (*tau*).

ADEV has a few disadvantages that are addressed by other deviation types, such as inability to distinguish white PM and flicker PM noise. It also exhibits relatively low confidence in the presence of smaller sample sizes, compared to some newer statistics. Still, ADEV is among the most commonly-used performance metrics for high-performance clocks and frequency standards.

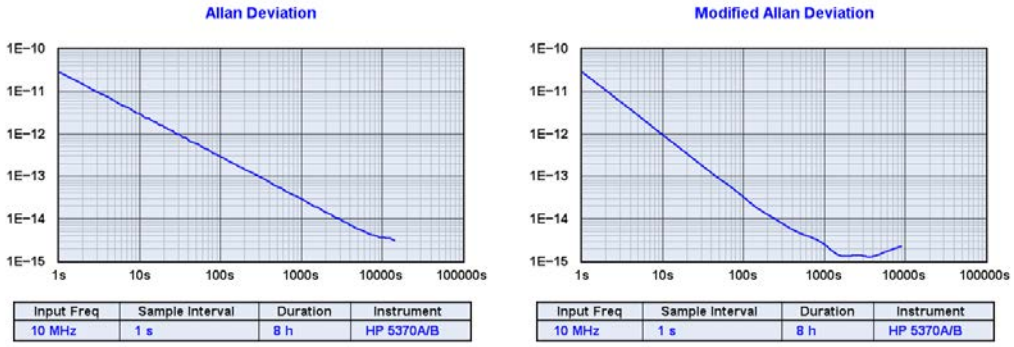
Modified Allan Deviation (m)

MDEV, or Mod $\sigma_y(\tau)$, is a slight variation on ADEV that can discriminate between white PM and flicker PM noise. Both ADEV and MDEV render flicker PM noise at a slope of τ^{-1}. White PM noise is also rendered by ADEV at τ^{-1}, but it appears as a steeper downward slope of $\tau^{-3/2}$ in MDEV. Consequently an ADEV plot may appear artificially elevated when either of these two noise types is present. Switching to **Measurement→Modified Allan**

[4] W. J. Riley, *Handbook of Frequency Stability Analysis* (http://tf.nist.gov/general/pdf/2220.pdf)

Deviation (m) will render the two noise types separately, at the timescales where they belong.

As an example, here are ADEV and MDEV plots showing the residual noise of an HP 5370B time interval counter. For this test, the counter's START input was fed with a 1-pps divider, which was driven by a 10 MHz clock that was also connected to the STOP input using an RF splitter. The counter's noise floor consists almost entirely of white PM and flicker PM noise.



The ADEV plot shows a consistent τ^{-1} slope to 10000 seconds, which could be indicative of either white PM or flicker PM noise. The MDEV plot renders the white PM noise at a slope of $\tau^{-3/2}$, transitioning to flicker PM near τ=400s or so. Limited sample availability beyond τ=2000s makes it difficult to draw further conclusions, but it's clear that MDEV reveals more about the counter's noise characteristics than ADEV.

Hadamard Deviation (h)

HDEV, expressed as $H\sigma_y(\tau)$, can be thought of as a 3-sample alternative to ADEV. HDEV plots of drift-free sources will generally appear similar to ADEV, but while ADEV fails to converge in the presence of linear drift, HDEV is unaffected by it. This means that switching from ADEV to HDEV will yield results similar to viewing ADEV after an **Edit→Subtract global linear frequency trend (drift line) (Ctrl-l)** operation. (The results will not be identical because HDEV also responds to some high-divergence noise types.)

One interesting use for HDEV is predicting the ADEV performance of crystal oscillators that have not yet fully warmed up. Unless the oscillator is exhibiting frequency jumps or inconsistent drift characteristics, its short-term ADEV will eventually resemble the HDEV performance that was observed only a few minutes after power-up.

Time Deviation (t)

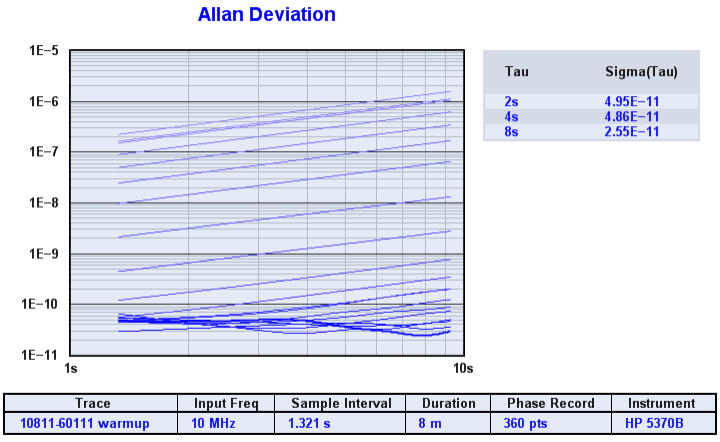
TDEV, or $\sigma_x(\tau)$, expresses the time stability of phase at the specified tau, in units of seconds. Numerically, TDEV is equal to (MDEV * τ) / sqrt(3). It's similar to the TIE (Time Interval Error) statistic used by the telecommunications sector.

Examining changes in stability over time

To make basic "dynamic ADEV" measurements[5], refer to the help text for the **Duration** and **Run Until** fields in the acquisition dialog, as well as the **Trace History** field. (In the TimePod dialog, **Trace History** is found on the **Frequency Stability** tab.)

Specifying a **Trace History** value greater than 1 causes TimeLab to divide the phase record into the given number of regions. Subsequent displays of xDEV measurements will assign each region its own trace. As the measurement runs, more recently-acquired regions towards the end of the phase record are drawn in darker colors, revealing how the stability of a given device changes over time. (If **Trace→Toggle trace thickness (T)** is enabled, only the most recent region will be displayed with a heavy trace.)

You can specify a new Trace History value at any time after acquisition in the **Edit→Trace properties (e)** dialog. In the example below, eight minutes of frequency readings were taken from an OCXO shortly after power-up, and later rendered with **Trace History** set to 20. Consequently, each ADEV trace represents about 24 seconds of phase data. As the oscillator warms up, the later traces exhibit better stability.



Values displayed on the *Sigma(Tau)* chart apply to the most recent trace.

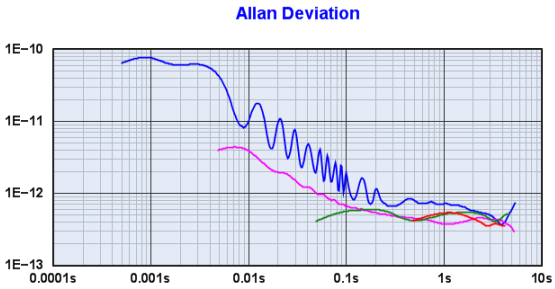
For more advanced displays of dynamic AVAR/ADEV and related metrics as described in the literature, consider using the **File→Export phase data to Stable32 (Ctrl-x)** command described on page 66 for offline analysis.

[5] Galleani, L. , Tavella, P.; Interpretation of the dynamic Allan variance of nonstationary clock data, *Frequency Control Symposium, 2007 Joint with the 21st European Frequency and Time Forum. IEEE International,* pp. 992-997, 2007

Common artifacts in ADEV and related measurements

High-resolution plots captured by the 5330A can reveal artifacts that don't seem to be present when the same measurement is made by other instruments. For example, spurs due to AC power coupling or ground loops, discussed below, appear very different when acquired at high sample rates and rendered with 20 or more ADEV bins per decade. Line-related spurs may not have been resolvable before, but that doesn't mean they weren't present!

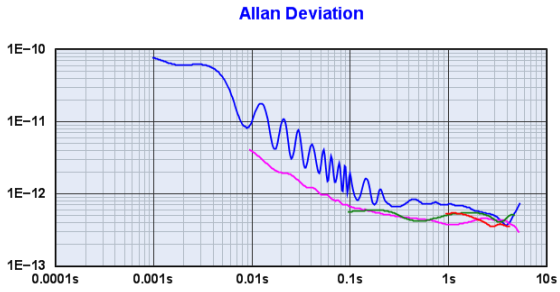
Measurement bandwidth and τ_0



Trace	Notes	Sample Interval	ENBW	Instrument
GPS clock	OCXO	0.000486 s	514.1 Hz	TimePod 5330A
GPS clock	OCXO	0.004 s	52.7 Hz	TimePod 5330A
GPS clock	OCXO	0.047 s	5.2 Hz	TimePod 5330A
GPS clock	OCXO	0.473 s	0.5 Hz	TimePod 5330A

One artifact that *does* originate in the 5330A's driver software is shown in the plot at right. Four successive ADEV acquisitions are shown, each conducted at a different noise-equivalent measurement bandwidth (ENBW) setting. Near the beginning of each trace, some "droop" is present that doesn't accurately reflect the behavior of the DUT.

What causes this artifact? Stein[6] has shown that the optimum ENBW for a given ADEV τ_0[7] is simply the $1/(2 * \tau_0)$ Hz Nyquist rate. This bandwidth is unachievable with a non-ideal antialiasing filter, so the 5330A driver internally acquires oversampled phase data that corresponds to an artificially short τ_0 interval relative to the selected ENBW. If allowed to appear on an ADEV plot, the antialiasing filter's response is easy to mistake for a real effect.



Trace	Notes	Sample Interval	ENBW	Instrument
GPS clock	OCXO	0.000486 s	514.1 Hz	TimePod 5330A
GPS clock	OCXO	0.004 s	52.7 Hz	TimePod 5330A
GPS clock	OCXO	0.047 s	5.2 Hz	TimePod 5330A
GPS clock	OCXO	0.473 s	0.5 Hz	TimePod 5330A

TimeLab supports two different ways to avoid displaying invalid data near τ_0. First, **Trace→Clip xDEV traces by noise bandwidth (**Ctrl-b**)** is enabled by default. (This option was turned off to demonstrate the issue above.) With noise bandwidth clipping enabled, as seen in the figure at left, TimeLab does not render any portion of an xDEV trace at taus shorter than 1/(2 * ENBW) seconds.

[6] *The Allan Variance – Challenges and Opportunities*, Samuel R Stein, Symmetricom, Inc., Boulder, Colorado USA (White paper available at www.symmetricom.com)

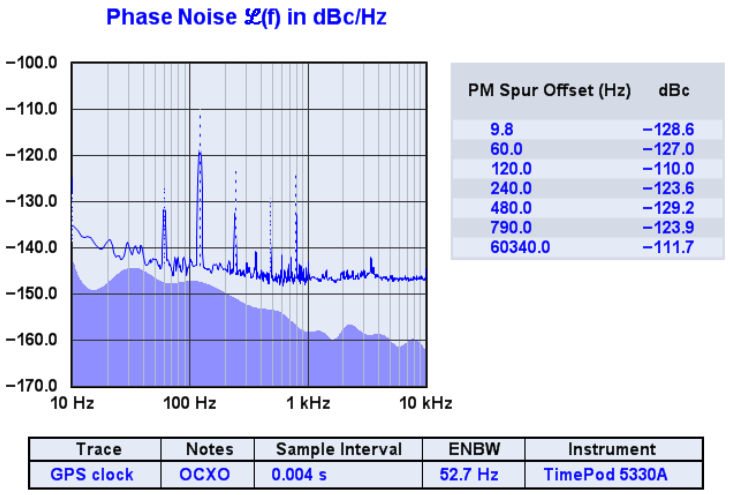
[7] τ_0 refers to the very first tau at the left end of an xDEV trace, which typically corresponds to the period of the phase or frequency samples being analyzed.

Additionally, beginning with version 1.1 of TimeLab, the TimePod 5330A driver can decimate the phase-difference data it acquires. Readings are collected by sampling the internal phase-difference data stream at a rate determined by the **Output Decimation** field in the acquisition dialog. This feature is primarily for convenience. It can be used to reduce the size of phase records generated by very long acquisitions, for example, or when high data rates must be used in order to track drifting sources. But decimation factors of 2:1 or greater also have the effect of satisfying the $\tau_0 >= 1/(2 * ENBW)$ criterion. With the default **Output Decimation** value of 2:1, phase-difference data that's affected by lowpass attenuation will be kept out of the measurement, even if **Trace→Clip xDEV traces by noise bandwidth (**Ctrl-b**)** is turned off.

Note that **Trace→Clip xDEV traces by noise bandwidth (**Ctrl-b**)** will have no effect on plots rendered with data captured from counters and other instruments that don't report their measurement bandwidth. It should be left enabled in most cases.

Line-related spurs

The most conspicuous artifact in the ADEV plots above is the "ripple" in the **blue** and **magenta** traces. This is caused by AC line interference, primarily a single 120-Hz spur:



Note that 1/120 Hz = 0.008s, which corresponds to the first (genuine) trough in the ADEV traces above. Depending on how they're introduced to the measurement, AC line spurs commonly appear at either the fundamental power-line frequency or its second harmonic. Users in the EU and other locales with 50 Hz power will see similar effects near τ=0.010 and 0.020s. In all of these cases, the device's "true" ADEV is close to an imaginary line drawn through all of the lowest points in the trace.

Getting rid of AC line spurs can be a challenge. Shielding at 50/60 Hz is usually impractical; if the interference is magnetically coupled into the measurement, there is

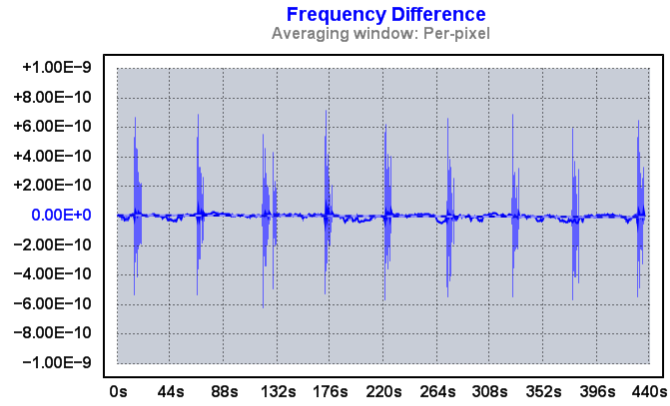
probably no cure besides identifying the offending source(s) and moving sensitive equipment and cables away.

Ground loops are a more likely offender, since all ports on the 5330A have ground pins or shields that are bonded to the instrument's metal enclosure. If you can identify the offending loop, you may be able to break its RF path with a coaxial balun[8].

You can also attack the source of the loop by bonding all equipment to a common ground that's connected to the building's power distribution network in only one place. To do this, ensure that all devices participating in the measurement – DUT, reference, TimePod, computer, and everything connected to them – are all part of the same AC circuit, preferably a power strip that allows all equipment to be plugged into a single outlet.

Using battery power can help, but batteries shouldn't be your first resort. Less drastic solutions can usually be found that will let you use the available AC power. (Try switching to MDEV, for instance.) If you are measuring stable sources with sufficient warmup time, you can eliminate line spurs by select the 5 Hz or 0.5 Hz **Measurement BW** option in the acquisition dialog.

Other environmental hazards



Trace	Input Freq	Sample Interval	Instrument
REF FTS4050 INPUT Z3801	10 MHz	0.050 s	TimePod 5330A

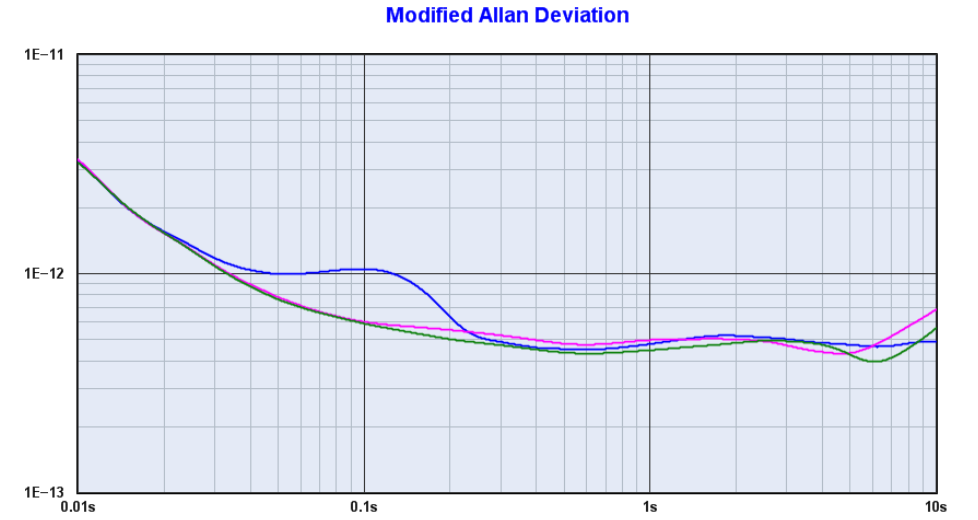
Above, a measurement of two 10 MHz sources was interrupted by bursts of apparently-random noise at roughly 50-second intervals. The culprit turned out to be a cellular phone sitting on a nearby bench. Try to keep all RF radiation sources – intentional and otherwise! – well away from your testing area.

[8] E.g., http://www.minicircuits.com/pdfs/FTB-1-1+.pdf. When using coaxial baluns, make sure they're actually helping, though. In particular, watch out for elevated phase noise floors, crosstalk artifacts, and assorted spurious responses picked up by the now-ungrounded coax shield.

Crosstalk

Finally, don't confuse power-line spurs or other sources of low-frequency interference with *crosstalk*. This is an insidious artifact that may be observed in high-performance measurements – or worse, may go unnoticed.

When working with the 5330A, crosstalk tends to appear when the signal frequencies at the **INPUT** and **REF IN** ports are nearly the same, but not precisely. As the two signals' phases slowly approach, coincide, and separate, any coupling between them creates a spurious response at the difference frequency. Appearing at a similar magnitude in both AM and phase noise plots, these spurs also cause peaks and nulls in ADEV and related measurements.



Trace	Input Freq	Ref Freq	Sample Interval	ENBW	Instrument
Morion MV89A, 3m RG58, house clock on	10.000003 MHz	10 MHz	0.004 s	52.7 Hz	TimePod 5330A
Morion MV89A, 3m RG58, house clock off	10.000003 MHz	10 MHz	0.004 s	52.7 Hz	TimePod 5330A
Morion MV89A, 3m RG400, house clock on	10.000003 MHz	10 MHz	0.004 s	52.7 Hz	TimePod 5330A

Crosstalk can also occur when a *third* signal, incidental to the measurement environment, causes beatnote effects that are asynchronous to both test inputs. This case is illustrated in the plot above, where a TimePod was used to measure a high-stability OCXO against another one of similar quality.

In this case, while the OCXO driving **REF IN** was very close to 10 MHz, the OCXO driving the **INPUT** port was set approximately 3 Hz higher for demonstration purposes. Also present in the room (but not participating in the measurement) was a GPS-disciplined frequency standard and distribution amplifier supplying 10 MHz to various instruments through long cables.

The 5330A's reference source was connected with a double-shielded RG400 cable in all three trials. The **blue** trace and **magenta** traces were taken with the input source connected with a 3m length of RG58. A significant MDEV peak was visible in the **blue**

trace… but when the GPSDO was powered down for the **magenta** trace, the spurious response vanished.

Likewise, as the **green** trace shows, replacing the RG58 input cable with an equal length of RG400 also eliminated the interference from the house standard.

As can be seen, the recommendation to use double-shielded cables with the TimePod is not born of excessive caution. Even plots acquired by conventional frequency and TI counters are vulnerable to crosstalk! With the widespread availability of GPSDOs and other inexpensive frequency standards, many facilities from international physics laboratories to ham shacks are equipped with 5- and 10-MHz distribution networks. Don't use low-quality cables with high-performance instrumentation.

Hints for xDEV measurements

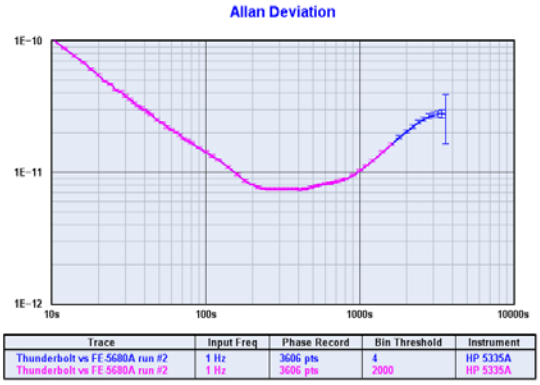
- The TimePod 5330A does not display its own measurement floor in the ADEV, MDEV, HDEV, or TDEV views. Instead, you can (and should) make residual plots for measurements that may approach the instrument floor, using similar frequencies and signal levels. These residual plots can be saved, loaded and shown with **Display→Overlay all loaded plots (o)** next to your actual measurement traces.

- There's a direct relationship between measurement bandwidth (as selected on the **Frequency Stability** tab of the TimePod acquisition dialog) and the TimePod's measurement floor in ADEV and other phase/frequency measurements . You may find it helpful to observe drift-prone sources or oscillator startup characteristics at 500 Hz ENBW, but be aware that the noise floor will be higher. Again, a separate residual plot at the frequencies and signal levels of concern can tell you exactly where the limits are.

- The measurement bandwidth also influences the size of the phase record that must be allocated before a measurement of a given duration can begin. As a result, you may experience out-of-memory errors if you tell TimeLab to record several hours' worth of phase data at 500 Hz ENBW/1K points per second. Increasing the **Output Decimation** value in the **Frequency Stability** tab is a great way to reduce acquisition memory requirements, but it will not change the xDEV noise floor. The only way to lower the xDEV floor is to select a narrower measurement bandwidth.

- Fewer samples are available to contribute to bins near the right end of an xDEV trace. As a result, measurement confidence decreases at longer tau intervals. To increase the minimum number of samples that must contribute to a given xDEV bin in order for that bin to be displayed, you can change the **Bin Threshold** parameter in the acquisition dialog, or use **Edit→Trace properties (e)** to specify a new value for the same field at any time after acquisition.

  However, because TimeLab employs overlapped xDEV algorithms, small changes to **Bin Threshold** may have no effect. Once the phase record grows long enough to provide data at a given tau interval, each additional sample adds another data point to the corresponding xDEV bin. If this is an issue, try setting the **Bin Threshold** to a large fraction of the total phase record size. The idea is to force TimeLab to render the xDEV trace as if much less data were available.



In the example at left, two copies of the same 3600-sample .TIM file have been loaded. The **blue** trace was rendered with the default **Bin Threshold** of 4, while the shorter **magenta** trace was rendered with **Bin Threshold** set to 2000. Enabling **Trace→Show xDEV error bars (**Ctrl-e**)** reveals the **blue** trace's low confidence at its longest tau.

Generally, as long as **Trace→Clip xDEV traces by confidence (**Ctrl-v**)** is enabled, it will not be necessary to alter the **Bin Threshold** value to display xDEV trace data of good statistical quality. The default bin threshold and clipping options will suffice for almost all measurements.

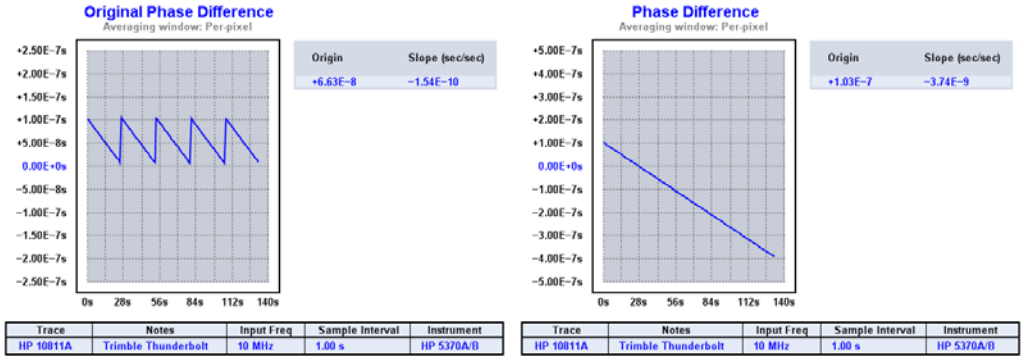
Working with phase- and frequency-difference traces

Phase and frequency stability data acquired by TimeLab is represented internally as an array of phase-difference samples. This is true regardless of whether the data came from a frequency counter, a time-interval counter, or a direct-digital timing analyzer. (Frequency readings are converted to phase-difference samples on the basis of their deviation from the first frequency reading acquired.)

In addition to the statistical measurements such as Allan deviation that TimeLab provides, "strip chart" views are available to give you a closer look at the raw phase-difference data. These views correspond to the **Phase difference (Unwrapped) (p)**, **Phase difference (Original) (w)**, and **Frequency-difference (f)** options on the **Measurement** menu.

Phase difference (Original) (w)

Phase difference (Unwrapped) (p)



The distinction between these two measurement types is relevant when receiving time-interval data from a traditional counter. The **Phase difference (Original) (w)** measurement shows the original TI samples as they arrive from the hardware. When acquired from a TI counter, these readings are subject to "wraparound" when they exceed the period of the signal that determines the START-to-STOP channel interval. For instance, if an oscillator near 10 MHz is measured against a drift-free source at exactly 10 MHz, the TI readings from the counter will increase or decrease from one trigger interval to the next. When the START-to-STOP interval exceeds 100 nanoseconds or falls below 0 nanoseconds, the TI readings will necessarily wrap back around to the opposite "rail." This phenomenon gives rise to the sawtooth-shaped phase trace familiar to users of vector network analyzers. As seen in the measurement example above, the phase-wrapping effect is visible in the **Phase difference (Original) (w)** measurement view in TimeLab.

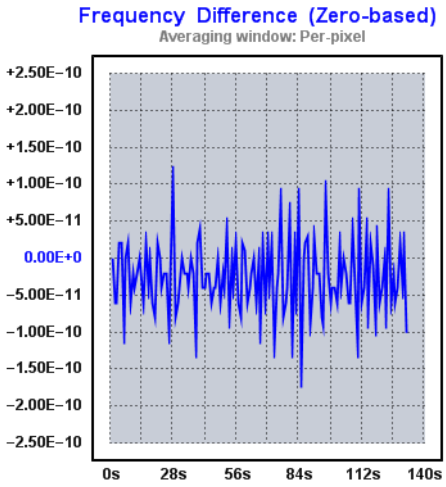
Conversely, TimeLab's **Measurement→Phase difference (Unwrapped) (p)** view displays the TI samples in the form of phase data that's been "unwrapped." The sawtooth

discontinuities are removed by adding or subtracting the input signal's period whenever the sample-to-sample time difference exceeds half of that period.

It's seldom necessary to refer to the original phase-difference graph except when troubleshooting configuration problems with live TI counter measurements. In most cases you should select the **Measurement→Phase difference (Unwrapped) (p)** view when you wish to inspect the principal phase record for your acquisition. Additionally, all of the flattening, detrending, and data-removal commands on the **Edit** menu work by replacing the original phase data with modified data based on the unwrapped record. After executing any of these commands, the **p** and **u** views will always be identical, just as if the data had been acquired from a frequency counter or direct-digital analyzer.

Further references to the phase-difference view in this manual will appear simply as **Measurement→Phase difference (p)**, omitting the distinction between wrapped and original phase data.

Frequency difference (f)



Origin	Drift (Hz/sec)	Drift (Hz/min)
−2.57E−11	−8.04E−8	−4.82E−6

Avg Time (s)	Freq (Hz) at 132s	Error
1	9 999 999.961 900 000	−3.81E−9
3	9 999 999.962 500 000	−3.75E−9
10	9 999 999.962 619 999	−3.74E−9
30	9 999 999.962 636 666	−3.74E−9
100	9 999 999.962 634 999	−3.74E−9

Trace	Notes	Input Freq	Sample Interval	Instrument
HP 10811A	Trimble Thunderbolt	10 MHz	1.00 s	HP 5370A/B

Shown above is a third way to view the phase data from the earlier example. Each point on the **Measurement→Frequency difference (f)** trace is calculated by dividing the difference between successive pairs of phase-difference samples by the sample rate. Taking the time derivative of *n* adjacent phase-difference points in this manner yields a stream of *n-1* frequency-difference points. Because of this differentiation process, the **Measurement→Frequency difference (f)** view of the phase record is often better at revealing glitches and other short-lived artifacts in the phase record than the actual

phase-difference trace is. Long-term oscillator drift and temperature-dependent effects are also easily spotted.

When the optional chart to the right of the trace area is enabled with **Display→Numeric table (Ctrl-n)**, TimeLab can be used to emulate a frequency counter with impressive performance. The chart can be viewed at any time during or after acquisition. Its entries are computed in real time by integrating the frequency-difference values in the currently visible trace at successively longer intervals. The integration process moves from right to left, starting with the newest sample points and moving farther back in history as more data becomes available.

How does TimeLab measure frequency?

It's important to understand that the "frequency differences" shown on the graph are actually fractional deviations from a value known as the *input frequency*. If the measurement's input frequency is exactly 10,000,000 Hz, then a frequency-difference reading of +1E-7 would mean that the absolute frequency is 10,000,001 Hz at that moment in time.

But what exactly is this "input frequency?" Where does it come from, and how accurate is it? When acquiring data with a frequency counter, it's easy enough to determine the nominal input frequency for the measurement: we simply use the first reading. All subsequent data points in the phase record are computed using the difference between the incoming frequency readings and the first one recorded… and all of these readings can be assumed to carry the same accuracy and precision.

When a time-interval counter (TIC) is used, however, TimeLab sees only a series of START-to-STOP interval times. It's impossible to infer the input frequency automatically. Instead, you must enter it into the appropriate field of the acquisition dialog, specifying at least as much precision as you expect to obtain when viewing the frequency count chart.

Phase/frequency measurements with the TimePod

For TimePod 5330A users, the situation is a bit more complex. First it must be emphasized that the values on the **Measurement→Frequency difference (f)** plot have nothing to do with the difference between the *absolute* signal frequencies at the **INPUT** and **REF IN** jacks, as might be surmised. Instead, the frequency-difference plot reveals *changes* in the difference between these frequencies that occur after measurement begins.

On the 5330A, the process of determining the input frequency can be thought of as a choice between the frequency-counter and TIC methods. Options in the **Acquire→Miles Design TimePod** acquisition dialog allow you to enter an explicit **REF IN** frequency, both **INPUT** and **REF IN** frequencies, or neither. The choice between ultimate accuracy and "plug and play" convenience is left up to you.

Measurement initialization

To understand the compromises involved, let's look at some of the steps that the 5330A software driver executes when you press the **Start Measurement** button. Refer to the figure above, taken from the **Acquire→Miles Design TimePod** acquisition dialog.

1) Using the nominal ADC clock frequency specified in the acquisition dialog -- typically 78.050 MHz or 76.050 MHz -- zero-crossing detectors are used to obtain a rough estimate of the signal frequencies at the **INPUT** and **REF IN** jacks.

2) If you unchecked the **Measure** option next to the **Reference Freq** field and entered an explicit reference frequency, this value replaces the estimated **REF IN** frequency. The difference between the estimated and entered reference frequencies is used to improve the estimate of the ADC clock frequency.

3) If you unchecked the **Measure** option next to the **Input Freq** field and entered an explicit input frequency, this value replaces the estimated **INPUT** frequency. (The **INPUT** frequency may be specified only if **REF IN** is also specified explicitly.)

4) The DDS cores on the FPGA are tuned to the estimated or entered input and reference frequencies.

5) The driver spends a few seconds measuring the actual input- and reference-channel baseband signals. For any frequencies you didn't specify explicitly, these baseband measurements are used to refine the current estimates. The DDS cores are then fine-tuned using the updated input, reference, and clock frequencies.

6) If an explicit reference frequency was specified, the reference channel's baseband frequency is measured once again. It should be very close to DC unless the ADC clock or the reference is drifting. If the reference-channel baseband frequency is greater than 5% of the selected measurement bandwidth, the observed difference is used to refine the ADC clock frequency estimate, under the assumption that an explicitly-specified reference frequency is likely to be much more stable than the 5330A's own clock. The driver returns to step (5) above in this case, looping indefinitely until the baseband signals are close enough to DC.

As the measurement begins, the **Input Freq** and **Ref Freq** fields in the legend table beneath the graph will be brought up to date based on the progress made through the steps above. Any estimated frequencies are displayed at 1-kHz precision, while any explicitly-specified frequencies are displayed at the precision given.

Some of the implications are rather subtle. For example, if you don't enter an explicit reference frequency, the accuracy of your frequency-count chart entries will depend on how close the actual reference frequency is to the 1-kHz rounded estimate used by TimeLab. This means that if your **REF IN** jack is connected to an atomic or GPS standard whose output is nominally a multiple of 1 kHz, as is often the case, then the accuracy of your frequency-count chart entries will be as good as the reference itself.

How does this work? The 5330A's internal estimate of the input frequency may not be very accurate, but it doesn't need to be. The slope of the phase differences measured by the 5330A is always equal to the frequency offset between the input and reference signals at baseband, regardless of how accurately or inaccurately their original RF frequencies were known when the measurement began. To the extent that the reference's actual frequency matches the rounded estimate used by TimeLab, the frequency of the RF signal at the **INPUT** jack can be recovered simply by adding this offset. Once this is done, the frequency-difference graph points can also be scaled to conform to the **Input Freq** value that's displayed in the legend table. (**Input Freq** is also rounded to the nearest 1 kHz by TimeLab, if not specified explicitly.)

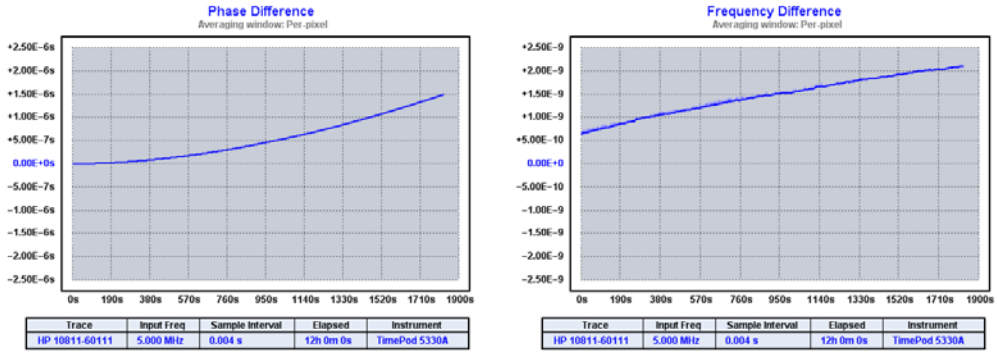
There's one small catch: although the input frequency will be reported accurately in the frequency-count chart as long as the rounded **REF IN** frequency is correct, the overall slope of the **Measurement→Phase difference (p)** trace may be inaccurate unless you specify both the input *and* reference frequencies in the acquisition dialog[9].

In particular, two-port phase stability measurements often require very high phase-slope precision, as do measurements in which the 5330A's own residual phase drift is tested. You should enter the common test-source frequency into both the **Input Freq** and **Reference Freq** fields when making such measurements.

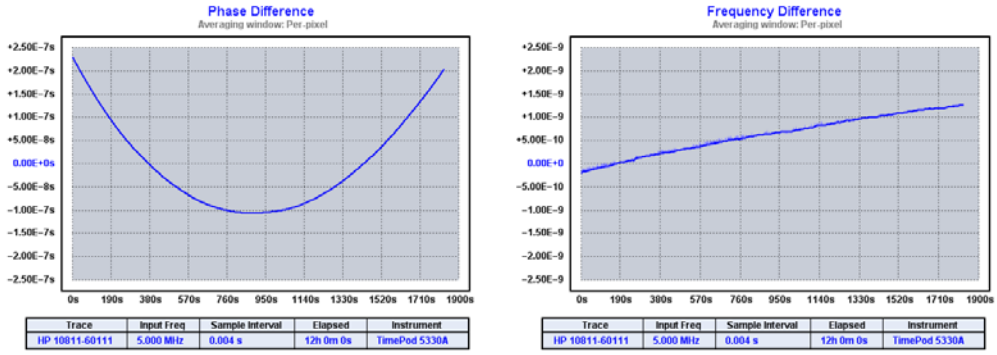
[9] Since the phase-difference slope reflects any offset in frequency between the downconverted input and reference signals, both frequencies must be accurately specified to the TimePod acquisition driver when making phase drift measurements at high precision. Even the slightest inaccuracy in estimation of the reference or input frequency during measurement initialization will add a nonzero bias to the baseband phase-difference slope.

Examining traces in detail

In the example below, a crystal oscillator was measured over a 12-hour period. The **Measurement→Phase difference (p)** and **Measurement→Frequency difference (f)** views appear side-by-side for comparison.



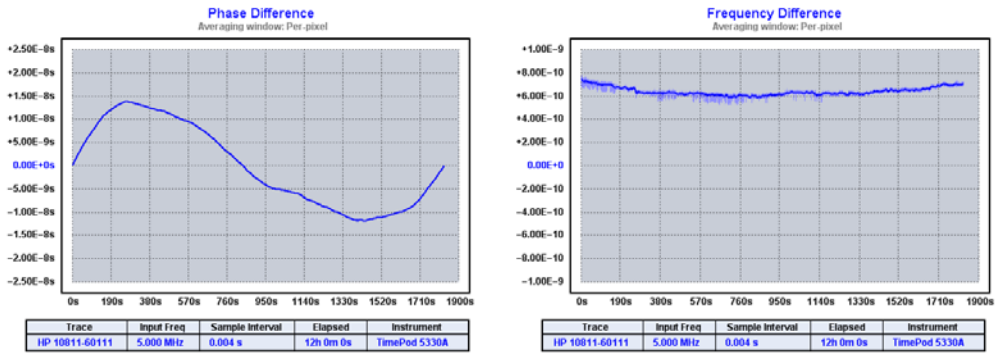
Not much detail is visible in either trace. Some sporadic frequency jumps are present, but they're not easily examined since their height is such a small fraction of the plot's overall magnitude. **Subtract global linear phase trend (frequency offset) (Ctrl-o)** command expands the phase-difference trace somewhat by removing its linear trend, but it only moves the frequency-difference trace down a bit:



In fact, we could have achieved a similar effect without altering the phase record by viewing the phase-difference trace with **Trace→Show linear phase/frequency residual (r),** or by viewing the frequency-difference trace with **Trace→Phase/frequency traces begin at zero (z)**.[10]

For these reasons, **Subtract global linear phase trend (frequency offset) (Ctrl-o)** is not a commonly-needed command. At first glance, **Subtract global linear frequency trend (drift line) (Ctrl-l)** seems more useful:

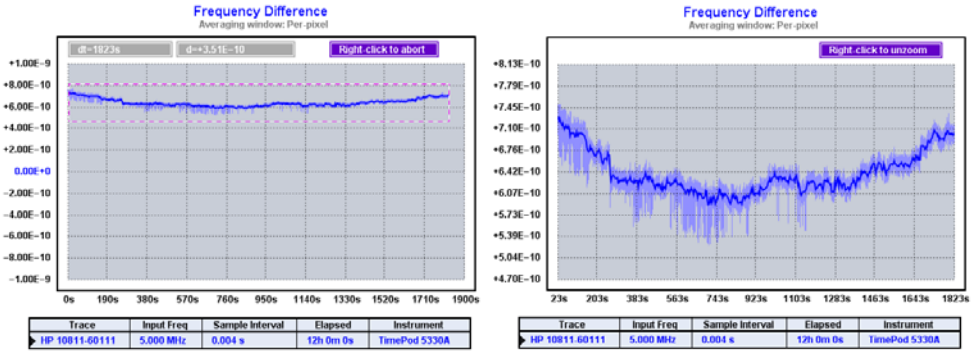
[10] The **Trace→Phase/frequency traces begin at zero (z)** option can actually be more useful – not only can it be toggled on and off without altering the phase data, but it also forces the beginning of the frequency-difference trace to midscale. See page 66.



Essentially, **Subtract global linear frequency trend (drift line) (Ctrl-l)** removes the quadratic trend from the phase record, which has the effect of removing the *linear* trend from the frequency-difference plot. But there's still an arbitrary offset in the frequency-difference view that's forcing the Y-axis scale to +/- 1E-9, and the artifacts we'd like to examine are much smaller than this.

Navigating zoomed graphs

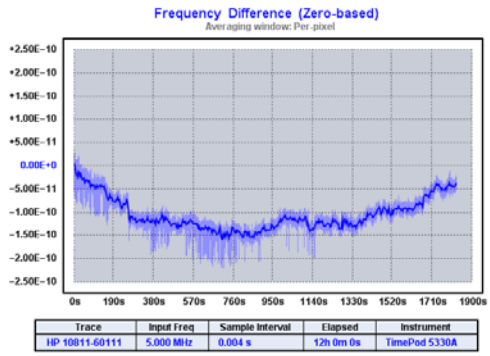
One strategy is to unlock the Y-axis with **Trace→Phase/frequency Y axis unlocked in zoom mode (y),** and then use the left mouse button to drag a box around the trace. When the button is released in the view at left, the result is the zoomed view at right:



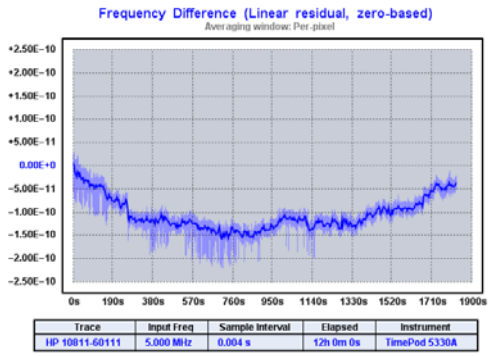
TimeLab is designed to take advantage of a PC mouse with three buttons and a scroll wheel. All of the graph types are "zoomable," and you can always return to the unmagnified view by right-clicking anywhere on the plot. Unlike the xDEV and noise measurement displays, though, the phase- and frequency-difference measurement views support additional navigation options. You can expand and contract the zoomed area with the mouse wheel, or surf through the data by dragging with the middle button.

Unlocking the Y axis with **Trace→Phase/frequency Y axis unlocked in zoom mode (y)** keeps TimeLab from constantly adjusting the Y axis to accommodate the graph's vertical range. With the Y axis unlocked, you can expand and scroll the graph in both directions with the mouse, rather than being limited to side-to-side navigation.

In this case, though, unlocking the Y axis and drawing a box requires a lot of user interaction. Worse, you'll end up with an "unfriendly" Y-axis scale whose random-looking values make the graph difficult to interpret. Once again, **Trace→Phase/frequency traces begin at zero (z)** offers a better alternative with a single keystroke:



To sum up, selecting **Measurement→Frequency difference (f)**, applying **Subtract global linear frequency trend (drift line) (Ctrl-l)** , and enabling **Trace→Phase/frequency traces begin at zero (z)** can reveal details which are impossible to spot in the raw phase-difference trace. All of this being said, in order to get the same detailed view, we *really* only needed to enable **Trace→Show linear phase/frequency residual (r)** while viewing the frequency difference trace!



The **Trace→Phase/frequency traces begin at zero (z)** and **Trace→Show linear phase/frequency residual (r)** commands are both extremely useful. Enabling one or both of these features can give you a detailed view of most phase- or frequency-difference traces without altering the phase data itself[11]. Keep the r and z keys in mind while working with these traces, and experiment with them frequently.

[11] Note that the **Subtract quadratic linear frequency trend (drift curve) (Ctrl-q)** operation has no "view-only" equivalent. It operates in a manner similar to the **Subtract global linear phase trend (frequency offset) (Ctrl-o)** and **Subtract global linear frequency trend (drift line) (Ctrl-l)** commands, and can be even better at revealing details in plots like this one.

Hints for phase/frequency stability measurements

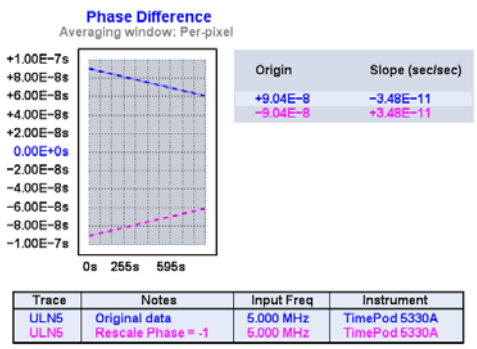
- Due to slight variations in FPGA/USB interface temperature that occur when acquisition begins, the 5330A's own impact on critical residual-phase measurements can be minimized if you use **Acquire→Enable deferred acquisition (ctrl-d)** to force the acquisition to throw away incoming data until you press Enter to 'trigger' it. This feature is normally used to improve synchronization in multiple acquisitions, but you can also use it to force a single instrument to reach its equilibrium temperature before the software saves any data.

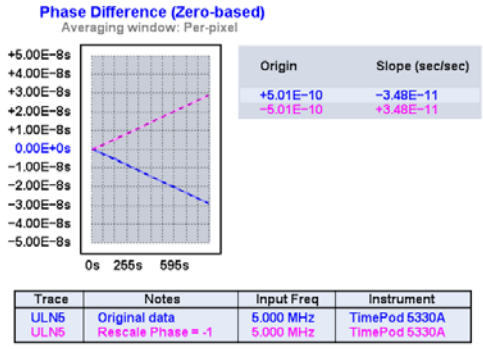
  You can accomplish the same thing by selecting the **Measurement→Phase difference (p)** view, using the left mouse button to select the first few minutes of data where the phase slope may be artificially degraded. When you release the button it will zoom to the selected area of the plot. You can then hit **F4** to delete that chunk of data from the phase record. The ADEV and other phase-derived graphs will be recalculated. (This will *not* affect the phase or AM noise plots; as mentioned elsewhere, they are acquired with a completely different "signal path" than the one that decimates and records the phase-difference data.)

- Pressing **Scroll Lock** will latch the current Y-axis limits, preventing them from changing regardless of the zoom status or **Trace→Phase/frequency Y axis unlocked in zoom mode (y)** option. This feature can be useful when you want to keep the view from being rescaled after removing, detrending, or flattening trace data, such as when similar modifications are being applied to multiple traces in an unzoomed view. Turning **Scroll Lock** off will restore the previous Y-axis control state.

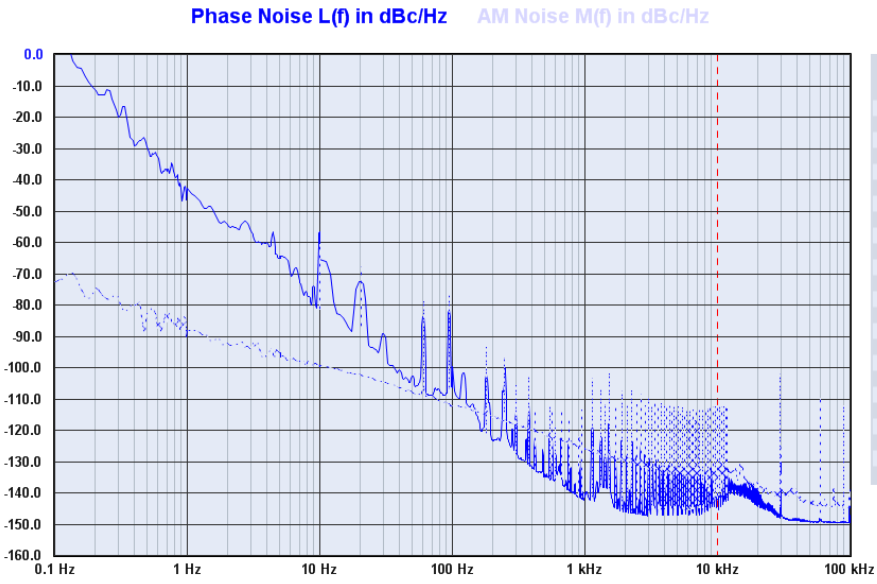
  Most users will not need to latch the Y axis limits. **Scroll Lock** is one of the few operations in TimeLab with no menu-based equivalent.

- To invert the phase slope, you can use the **Edit→Trace properties… (e)** dialog to specify a **Rescale Phase** value of -1. This will invert both the slope and the origin of the phase data, but you can still use **Trace→Phase/frequency traces begin at zero (z)** to display the inverted trace at a fixed origin of zero.





Phase noise, AM noise, and jitter



PM Spur Offset (Hz)	dBc
10.0	-60.6
20.0	-68.0
60.0	-78.9
94.0	-77.0
180.0	-93.7
250.0	-96.8
300.0	-105.7
380.0	-102.8
420.0	-112.9
540.0	-111.8
570.0	-113.6
600.0	-118.8
660.0	-113.7
750.0	-115.5
900.0	-122.9
940.0	-113.6
1130.0	-103.4
1320.0	-106.5
1420.0	-111.8
1520.0	-102.2
1700.0	-112.0
1890.0	-107.5
2070.0	-112.2
. . .	

Trace	Notes	Input Freq	Input Amplitude	Ref Freq	Ref Amplitude	dBc/Hz at 10 kHz	Instrument
HP 8640B (Unsaved)	HP 5061A	5098148.754 Hz	9.5 dBm	5000001.076 Hz	15.9 dBm	-140.8	TimePod 5330A

To measure the phase noise of a drift-prone source *such as this HP 8640B cavity-tuned signal generator, uncheck the* ***Frequency Stability*** *option in the acquisition dialog. This will allow greater drift tolerance, since it won't be necessary for the signal to remain within a narrow measurement-bandwidth passband*

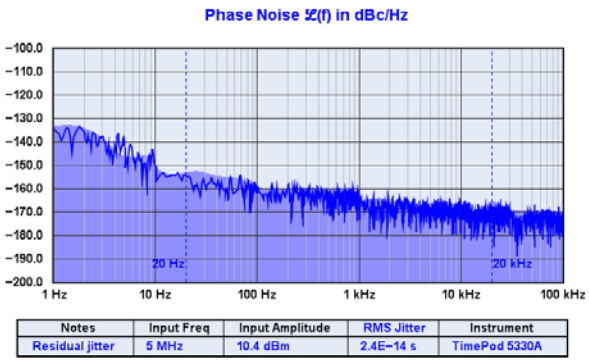
Thanks to the magic of cross-spectrum averaging, the TimePod 5330A can measure phase noise and AM noise on HF signals at exceptionally low levels. Below are some guidelines to help you make the most of this capability.

Integrated noise and jitter measurement

You can display various measures of integrated phase noise between two arbitrary offsets by enabling any or all of the following **Legend** menu fields:

- **Residual FM**
- **RMS Integrated Noise (Degs)**
- **RMS Integrated Noise (Rads)**
- **RMS Time Jitter**
- **SSB Carrier/Noise**

When one or more of these parameters is enabled for display in the legend table, two "spot cursors" will appear in blue. You can move these lower and upper limit cursors with **Ctrl-left click** and **Ctrl-right click**, respectively.

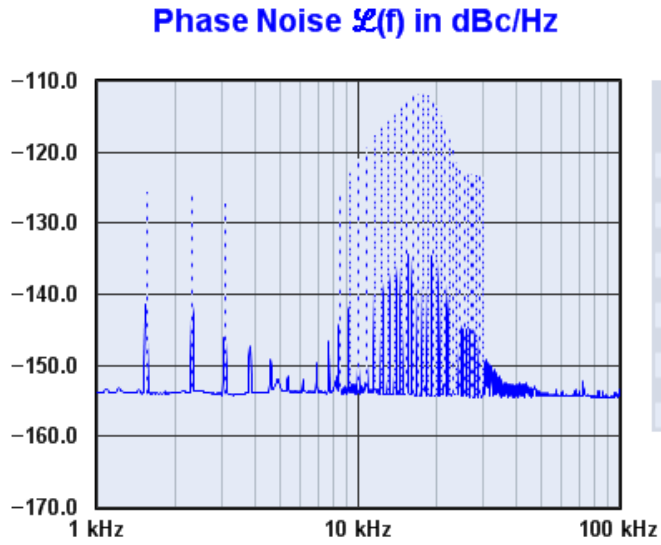


TimeLab does not make a distinction between random and deterministic jitter, so any spurs in phase noise plots are "integrated" as if they contained only random noise. Consequently, when viewing any of the RMS integrated noise values, you should use either **Trace→Smooth noise traces (**Ctrl-w**)** or **Trace→Suppress spurs (**Ctrl-s**)** to remove as much coherent energy as possible from the phase noise trace. Otherwise, time jitter and integrated-noise readings may be artificially exaggerated.

Along with many other sources, HP Application Note AN270-2 provides a concise summary of the math behind phase-noise integration.

The spur table

The *spur table* is a list of all coherent amplitude- or phase-modulated spectral components detected by the instrument in the course of measuring AM noise or phase noise, respectively. Each plot acquired from an instrument that supports spur detection has its own spur table. Spur levels are reported as CW power levels relative to the carrier, each at a given offset from the carrier frequency.



PM Spur Offset (Hz)	dBc
120.3	−115.3
241.6	−130.4
360.9	−135.8
480.2	−135.3
768.9	−125.2
1544.6	−125.8
2314.5	−126.1
3089.2	−127.3
8459.3	−125.3
9229.2	−123.0
. . .	

Typical phase noise measurement with accompanying spur table. *The default* **Spur Threshold** *value of 6 dB, specified in the* **Noise and Jitter** *tab of the TimePod acquisition dialog, ensures that artifacts of lower amplitude relative to neighboring areas of the trace are not classified as spurs. To illustrate the difference between coherent spur amplitudes and their apparent levels on the 1-Hz normalized noise trace,* **Trace→Suppress spurs in noise traces** (Ctrl-s) *is disabled and* **Trace→Mark spurs in noise traces** (Ctrl-m) *is turned on.*

Show or hide known spurs

Several commands in TimeLab determine how spurs are handled in phase noise and AM noise graphs.

Trace→Mark spurs in noise traces (Ctrl-m) draws shaded or dashed lines, depending on the **Trace→Toggle trace thickness for current measurement** (T) setting, for each detected spur on the AM or phase noise graphs. These lines reflect the amplitude of each spur as reported in the spur table, *which will not be the same* as the level of any corresponding bumps or spikes in the noise trace itself. Because phase noise and AM noise traces undergo normalization to 1 Hz bandwidth, the Y-axis labels don't reflect the true amplitude of any coherent

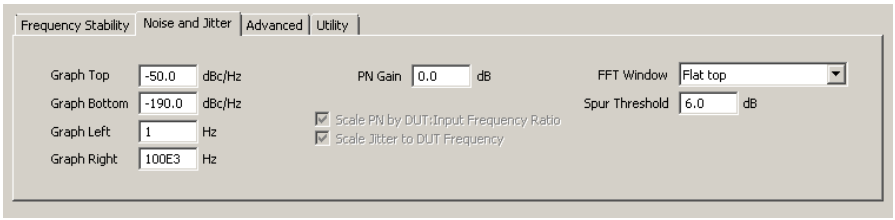
trace features. They must be recognized as spurs, and their amplitudes indicated separately.

Trace→Suppress spurs in noise traces (Ctrl-s**)** attempts to remove known spurs from phase noise and AM noise traces. As noted above, spurs that appear as artifacts in noise traces are not rendered at their true amplitude levels due to the use of FFT noise-bandwidth normalization. If you are concerned with spur amplitudes in a given measurement, you should use **Trace→Suppress spurs in noise traces (**Ctrl-s**)** to erase each detected spur from the graph and **Trace→Mark spurs in noise traces (**Ctrl-m**)** to replace them with vertical lines showing their true CW amplitude levels relative to the carrier.

Trace→Smooth noise traces (w**)** always implies **Trace→Suppress spurs in noise traces (**Ctrl-s**)**. For better visual quality, TimeLab will attempt to erase any known spurs prior to smoothing the trace.

Display→Numeric table (Ctrl-n) can be used to show or hide the spur chart for the selected plot. Like the numeric charts and tables associated with other measurement types, the spur chart appears to the right of the graph area. The Ctrl-n command toggles table visibility for all measurement types, not just the currently visible one.

Spur measurement options



The **Noise and Jitter** tab in the TimePod 5330A's acquisition dialog contains two parameters that influence how spurs are recognized and recorded. First, the **Spur Threshold** field defines the amplitude level, relative to the average level of nearby FFT bins, below which a given bin's amplitude will not be considered indicative of a coherent spur. Second, the choice of **FFT Window** implies a compromise between the accuracy at which a given spur's offset frequency and amplitude are reported. (The **Flat top** window excels at measuring spur amplitude levels, but the **Hann** window offers better frequency resolution, and may improve the odds of detecting a given spur as well.)

Finally, the **Edit segment table** button in the **Advanced** tab allows you to change the FFT kernel sizes used in the individual noise-trace segments, which can affect both spur-identification performance and acquisition time.

Is it a spur, or isn't it?

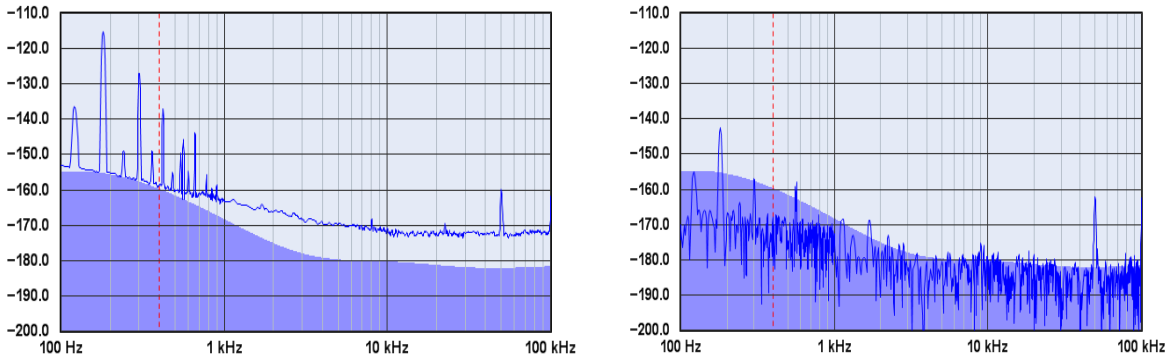
Identifying genuine spurs while disregarding false ones can be a challenge. The 5330A's DSP software may fail to detect spurs when they occur in clusters near the FFT segment's resolution limits, in steeply-sloped areas of the noise trace, or near segment boundaries. You can lower the **Spur Threshold** for a given measurement, but the risk of misclassifying random noise spikes as coherent spurs will become greater. When you see random spurs appear and disappear from the spur chart over the course of a measurement, you can increase the **Spur Threshold**, try changing FFT window types, or simply wait for the measurement to run long enough to reduce the trace variance ("grass").

Understanding instrument spurs

While TimeLab can sometimes overlook true spurs and report nonexistent ones, a third type of measurement error may need to be considered as well: the presence of spurs which originate within the instrument itself.

The spur-free dynamic range (SFDR) of the ADCs used in the 5330A is approximately 100 dB, which determines the corresponding specification limit of the instrument as a whole. In principle, your noise plots may contain fictitious spurs as high as -100 dBc, but the 5330A's typical performance is much better than this. Any spurs with amplitudes above -120 dBc that remain visible as the measurement converges are almost certainly "real" in the sense that they originate from the DUT, from the reference, or from environmental RFI.

Regardless of the equipment used, spur-free observations of AM noise and phase noise become challenging below -150 dBc. There are simply too many potential sources of coherent artifacts to rule out in most cases. Still, spurs that originate within the 5330A can sometimes be identified manually. An informal technique that may help determine whether a given spur or spur cluster originates within the 5330A hardware itself is illustrated in the side-by-side plots on the next page.



At left, the phase noise of a very high quality crystal oscillator has been measured over the course of several hours. Most of the visible spurs are well below the 5330A's specified limits, so it isn't clear which ones the instrument may be responsible for.

However, because complex signals are used throughout the 5330A's DSP pipeline, we can take advantage of a mathematical observation.[12] While the cross spectrum of a physically-realizable signal should fall on the real axis of the Argand plane, at least some instrument spurs are caused by data-conversion errors that can show up elsewhere in phase space. Keeping in mind that the 5330A achieves its low noise floor by cross-correlating the results reported by two independent "instruments" in the same box, these errors suggest some correlation between the noise from one ADC with the noise from its companion in the other "instrument." Like random ADC noise, the numerical distribution of these errors is not limited to the real axis… but unlike random noise, they don't average to zero over time. They appear on the graph as instrument spurs.

Switching to the **Trace→Show imaginary part of cross spectrum display (**Ctrl-F3**)** view at right reveals the noise and spurs that correspond to the imaginary part of the cross-spectrum average, specifically the absolute value of the Q-component at each bin. Ordinarily this data is used only to create the shaded area used by the **Trace→Show estimated instrument noise (**F2**)** feature, after undergoing heavy averaging. But Ctrl-F3 lets us view the imaginary part of the cross spectrum at full resolution. It's readily apparent that the spur near 50 kHz has *almost the same amplitude in the imaginary cross spectrum as it does in the real measurement.* This is a strong indicator that the spur is an instrument artifact.

There are other spurs in the imaginary cross spectrum that correspond to similar spurs in the real plot, but most appear to be AC line harmonics that both the 5330A and the sources under test are exposed to, and their amplitudes are substantially different. Although the Ctrl-F3 technique isn't foolproof, it can offer some valuable insight when you need to account for unanticipated low-level spurs.

[12] See Rubiola and Vernotte, The cross spectrum experimental method, and Grove *et al.*, Direct-digital phase noise measurement

Hints for noise measurements

- The 5330A can render an estimate of its own noise floor in AM/PM noise plots if you enable **Trace→Show estimated instrument noise (**F2**)**. See page 84 for more information.

- For AM and phase noise, a good indication that the noise trace has converged on its final level is the variance of the trace, or its overall fuzziness/thickness. If you measure a conventional signal generator or noisy oscillator, you'll notice that the trace stops falling relatively quickly after the measurement begins, and then becomes smoother over time. This 'smoothing' effect is an important cue that tells you how trustworthy your noise measurement is. With quiet sources, close-in segments can take an especially long time to converge at a final, smoothed level, since they are updated much less often.

- Crosstalk and AC power-line interference can create discrete spurs on phase noise and AM noise graphs, just as they can degrade ADEV and other statistical plots. See page 36 for a discussion of causes and cures.

- TimeLab has a graph-smoothing feature (**Trace→Smooth noise traces (**ctrl-w**)**) but it's often not needed with a cross-correlating analyzer such as a TimePod 5330A or a TSC 5120A/5125A. The best way to obtain a smoother trace with these instruments is to let your measurement run longer. Low-level residual plots can benefit from **Trace→Smooth noise traces (**ctrl-w**)**, but many measurements don't need additional smoothing at all.

- Another tip for smoother phase noise and AM noise traces, suitable for TimePod users with faster PCs: enable the **Overlapped Acquisition** option in the **Noise and Jitter** tab of the acquisition dialog. Overlapped acquisition provides better utilization of incoming data by the FFT routines, but requires significantly more CPU time.

- When overlaying several measurements on the screen, it can be helpful to run with spurs removed from the trace (**Trace→Suppress spurs (**Ctrl-s**)** enabled) and unmarked on the graph (**Trace→Mark spurs (**Ctrl-m**)** disabled).

TimeLab command reference

Most of the menu-based commands in TimeLab also have single-key shortcuts for faster, more interactive workflow. When present, the shortcut key is shown after the name of the menu option.

File menu

The File menu helps you work with data files and images. Supported actions include opening, saving, closing, and deleting plots stored in .TIM files, saving image files, printing or copying the currently displayed window contents, and importing and exporting raw measurement data. Various "user preference" toggles are also provided.

Load .TIM file (l)

Loads a saved .TIM file. .TIM files are ASCII text files that contain all measurement data acquired or imported by TimeLab, together with any applicable metadata. Specific fields and tables within each .TIM file may vary depending on the measurement type(s) saved in the file.

For frequency stability measurements made with counters or timing analyzers, the .TIM file contains the phase record obtained from the instrument during acquisition. The phase record may have been derived from a series of frequency readings or other raw data stream formats. It may also have been decimated to conform to a specified measurement rate or bandwidth.

For phase noise and AM noise measurements, the .TIM file contains the FFT bin contents needed to render the applicable graphs.

Save image or .TIM file (s)

This command can either save the selected plot as a .TIM file, or save the current contents of the TimeLab window as an image file in .PNG, .GIF, .TGA, .PCX, or .BMP format.

By default, .TIM files will be saved. To save an image file, simply add one of the suffixes above when you enter the filename. *Keep in mind that image files do not preserve your original measurement data, and cannot be reloaded into TimeLab.*

Generally, the .PNG format is best for saving images from TimeLab for email attachment or inclusion in other documents. .GIF and .PCX files will discard some color information, while .TGA and .BMP files are unnecessarily large.

When saving an image, all visible status and error messages, notifications, prompts, and mouse-cursor query values are removed. The black-triangle cursor associated with the currently selected plot will also be omitted from saved images.

Copy image to clipboard (Ctrl-c)

This command offers a handy shortcut for cutting/pasting TimeLab screen images into other programs via the operating system's clipboard, avoiding the need to save an image file.

When copying the screen image to the clipboard, all visible status and error messages, notifications, prompts, and mouse-cursor query values are removed. The black-triangle cursor associated with the currently selected plot will also be omitted from the copied image.

Use caution when pasting images from the clipboard into email messages. Some clients may encode them in uncompressed form, resulting in messages too large to email reliably. When in doubt, consider attaching a saved .PNG file instead.

Import .PNP phase noise data (N)

Imports a .PNP file saved by PN.EXE, a popular freeware phase-noise measurement application from the KE5FX GPIB Toolkit. Phase noise plots from various RF spectrum analyzers not otherwise supported by TimeLab may be imported from PN.EXE and rendered in TimeLab.

Import ASCII phase or frequency data (L)

Reads an ASCII text file containing phase-difference, frequency, frequency-difference, or timestamp data, one entry per line. The imported data is converted to a standard TimeLab phase record, and may be rendered, manipulated, and saved as a .TIM file.

As with other acquisition dialogs, TimeLab's data-import dialog provides detailed mouseover help text for all fields. Refer to this help text for usage information.

To "watch" a continuously updated data file, use **Acquire→Acquire from live ASCII file** instead.

Export ASCII phase data (x)

This command is supported only when viewing the selected plot's phase record contents using **Measurement→Phase difference (p)** or **Measurement→Frequency-difference (f)**. It will save an ASCII text file containing phase difference data in seconds, one entry per line, at 16 digits of precision to the right of the decimal point.

If a selected or zoomed area has been defined by dragging with the mouse, only that portion of the phase record will be exported.

By default, the exported phase data will be saved with the suffix .CSV for ease of reading into Excel and other spreadsheets. Exported data may be re-imported into TimeLab, if necessary, by the **File→Import ASCII phase or frequency data** command. Select "All files" as the file type if the data was not exported with the suffixes .TXT or .DAT, then select **Phase difference (sec)** as the file data type.

Optionally, each sample in the ASCII phase data file can be timestamped with its absolute MJD or the number of elapsed seconds since the beginning of the phase record.

Export ASCII frequency data (X)

This command is supported only when viewing the selected plot's phase record contents using **Measurement→Phase difference (p)** or **Measurement→Frequency-difference (f)**. It will save an ASCII text file containing absolute frequency readings in hertz, one entry per line, at 16 digits of precision to the right of the decimal point.

If a selected or zoomed area has been defined by dragging with the mouse, only that portion of the phase record will be exported.

By default, the exported frequency readings will be saved with the suffix .CSV for ease of reading into Excel and other spreadsheets. Exported data may be re-imported into TimeLab, if necessary, by the **File→Import ASCII phase or frequency data** command. Select "All files" as the file type if the data was not exported with the suffixes .TXT or .DAT, then select **Frequency (Hz)** as the file data type.

Optionally, each sample in the ASCII phase data file can be timestamped with its absolute MJD or the number of elapsed seconds since the beginning of the phase record.

Export binary phase data

This command is supported only when viewing the selected plot's phase record contents using **Measurement→Phase difference (p)** or **Measurement→Frequency-difference (f)**. It will save a binary file containing phase difference data in seconds, represented as a block of double-precision IEEE 754 values in little-endian (Intel) format.

If a selected or zoomed area has been defined by dragging with the mouse, only that portion of the phase record will be exported.

As with **File→Export ASCII phase data (x)**, the default file suffix is .CSV. This is usually inappropriate for binary data; you should specify a suffix appropriate to your application. Exported binary phase data may not be re-imported into TimeLab.

Export phase data to Stable32 (Ctrl-x)

This command is useful for transferring data from the selected plot's phase record directly into a third-party analysis application such as Stable32 from Hamilton Technical Services. Stable32 is a popular application for offline data analysis. It supports many statistical features and display options beyond the limited set implemented by TimeLab.

This command works by exporting the selected plot's phase record to a temporary file using the same functionality as the **File→Export ASCII phase data (x)** command. If a selected or zoomed area has been defined by dragging with the mouse, only that portion of the phase record will be exported.

When the command is issued, a dialog box allows you to specify the location of the Stable32 executable or other third-party program, as well as the command line to be passed to it. The command line may contain various predefined variables described in the dialog box's help text, including placeholders for the name of the temporary file and the data rate (tau zero) in samples per second. Upon pressing the **Launch Stable32** button, the temporary file will be generated and the specified program launched. TimeLab will then continue to run normally.

Temporary files created by TimeLab are cleaned up when TimeLab exits, so it's advisable to leave TimeLab running until after the third-party analysis application has exited.

Export xDEV trace

Available in the **Measurement→Allan Deviation (a), Measurement→Modified Allan Deviation (m), Measurement→Hadamard Deviation (h),** or **Measurement→Time Deviation (t)** views, this command creates an ASCII text file containing a *tau, sigma(tau)* value pair for each visible xDEV bin in the selected plot. One value pair is written per line; each value is rendered with 16 digits of precision to the right of the decimal point.

The number and distribution of xDEV bins may be specified prior to acquisition with the **Bin Density** field in the **Frequency Stability** tab of the acquisition dialog. Additionally, the selected plot's bin density may be adjusted after acquisition through the **Edit→Trace properties** dialog. See the help fields in these dialogs for details.

By default, the exported *tau, sigma(tau)* pairs will be saved with the suffix .CSV for ease of reading into Excel and other spreadsheets.

Export AM/PM noise trace

Available in the **Measurement→Phase noise (P)** and **Measurement→AM Noise (A)** views, this command creates an ASCII text file containing *offset, dBc/Hz* values corresponding to the visible noise trace for the selected plot. One value pair is written per line; each value is rendered with 16 digits of precision to the right of the decimal point.

One *offset, dBc/Hz* pair is written for each pixel column in the displayed graph. This means that if you zoom in on a portion of the phase noise or AM noise graph, only the trace data corresponding to the visible area of the selected plot will be exported. Conversely, if the graticule's X-axis boundaries extend beyond the minimum or maximum offset for the selected plot, no *offset, dBc/Hz* pairs outside the valid range will appear in the exported file.

By default, the exported *offset, dBc/Hz* pairs will be saved with the suffix .CSV for ease of reading into Excel and other spreadsheets.

Note that when **Show imaginary part of cross spectrum (Ctrl-F3)** is active, the **File→Export AM/PM noise trace** command will export the imaginary cross spectrum trace data rather than the normal AM noise or phase noise trace.

Export AM/PM spur table

Available in the **Measurement→Phase noise (P)** and **Measurement→AM Noise (A)** views, this command creates an ASCII text file containing *offset, dBc* values corresponding to entries in the spur table for the selected plot. One value pair is written per line; each value is rendered with 16 digits of precision to the right of the decimal point.

See "Spurs" on page 59 for more about spur detection and measurement.

By default, the exported *offset, dBc* pairs will be saved with the suffix .CSV for ease of reading into Excel and other spreadsheets.

Print image

After displaying a standard dialog box for printer selection and configuration, this command prints the current contents of the TimeLab window.

The printed image will not contain any visible status or error messages, notifications, prompts, or mouse-cursor query values. The black-triangle cursor associated with the currently selected plot will also be omitted.

Scale file dialogs by window size

When this option is enabled, common file dialogs in Windows XP will be centered in the TimeLab window and resized proportionally. In later Windows versions, the option has no effect. (Default: disabled)

Warn before exiting with unsaved plots

If enabled, this option causes TimeLab to prompt for confirmation before exiting if any plots have been acquired or edited but not saved. (Default: enabled)

Reset all parameters, options, and settings at next startup

This option causes a dialog box to appear when you exit from TimeLab, confirming that you would like to reset all options, settings, colors, and dialog fields to their default values the next time the program is launched. If confirmation is given, the .INI files associated with TimeLab and all of its instrument drivers will be deleted, forcing them to be recreated with default values when the program is relaunched. (Default: disabled)

If desired, the .INI files used by TimeLab and its instrument drivers may be copied, backed up, or edited manually. The location of the directory where TimeLab stores its .INI files varies depending on the operating system version. In Windows 7, these files are normally located at **c:\Users*<user name>*\AppData\Local\Miles Design\TimeLab**, also accessible at **%LOCALAPPDATA%\Miles Design\TimeLab**. In Windows XP, the .INI files are normally located in the hidden directory **c:\Documents and Settings*<user name>*\Local Settings\Application Data\Miles Design\TimeLab**. TimeLab does not store configuration data in the Windows registry or anywhere else besides its .INI files; only installation-related data is stored in the registry.

Close selected plot (Del)

Closing the selected plot frees the memory it occupies and removes it from the graph and legend table. If the plot is associated with a saved .TIM file, the file is not deleted.

After closing a plot, the next plot in the legend table, if any, becomes the selected plot. Up to 9 plots may be loaded into memory at once; to acquire, load, or import additional plots, you must close at least one.

Close all visible plots (Home)

Closes all loaded plots at once, removing them from the graph and legend table.

Delete selected plot's .TIM file (Ctrl-Del)

After prompting for confirmation, this command unloads the selected plot's data from memory and also deletes its associated .TIM file, if any.

After closing a plot, the next plot in the legend table, if any, becomes the selected plot. Up to 9 plots may be loaded into memory at once; to acquire, load, or import additional plots, you must close at least one.

Quit (q or Esc)

Exits from TimeLab. Depending on the setting of **File→Warn before exiting with unsaved plots**, a confirmation prompt may be issued.

Unless you hold the Shift key as the program exits, TimeLab will write all of its global settings and dialog fields (as well as a list of known **Legend** menu entries and currently loaded plots) to a file called **TIMELAB.INI**. These properties will be restored when the program is next launched. The "Shift-exit" feature can be handy if you have opened or closed various plots or changed any options or colors, but don't wish to retain the changes or otherwise lose your previous startup defaults.

Note that acquisition and file-import dialog field contents are not backed up in **TIMELAB.INI**. The most recently used dialog entries associated with these operations are stored in instrument-specific .INI files. The instrument driver updates its .INI file as soon as you select **Start Measurement** or otherwise initiate an acquisition or import operation, not when the program exits.

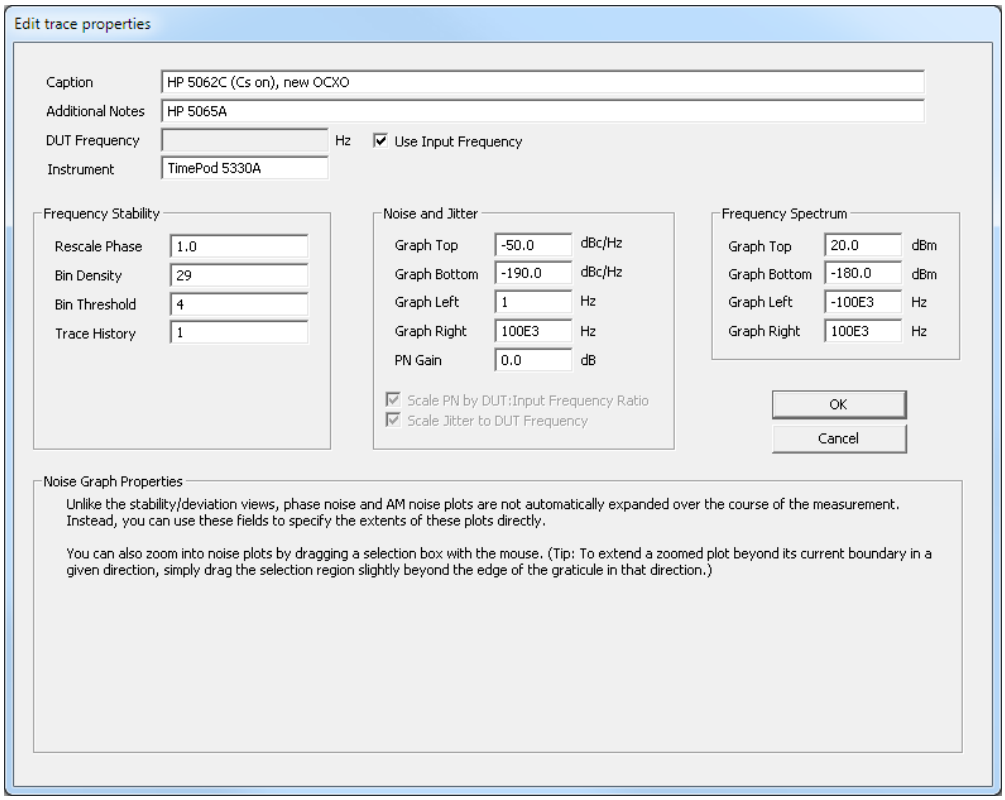
If desired, the .INI files used by TimeLab and its instrument drivers may be copied, backed up, or edited manually. The location of the directory where TimeLab stores its .INI files varies depending on the operating system version. In Windows 7, these files are normally located at **c:\Users*<user name>*\AppData\Local\Miles Design\TimeLab**, also accessible at **%LOCALAPPDATA%\Miles Design\TimeLab**. In Windows XP, the .INI files are normally located in the hidden directory **c:\Documents and Settings*<user name>*\Local Settings\Application Data\Miles Design\TimeLab**. TimeLab does not store configuration data in the Windows registry or anywhere else besides its .INI files; only installation-related data is stored in the registry.

After updating **TIMELAB.INI**, any temporary files that may have been created by instrument drivers or the **File→Export phase data to Stable32 (Ctrl-x)** command are deleted. For debugging purposes, temporary-file cleanup is also inhibited if you hold down the Shift key while the program terminates.

Edit Menu

Plots that have been loaded or acquired in TimeLab may be displayed with a variety of options and transforms. However, any changes to the measurement data or display parameters associated with a specific plot must be made through the commands in the Edit menu.

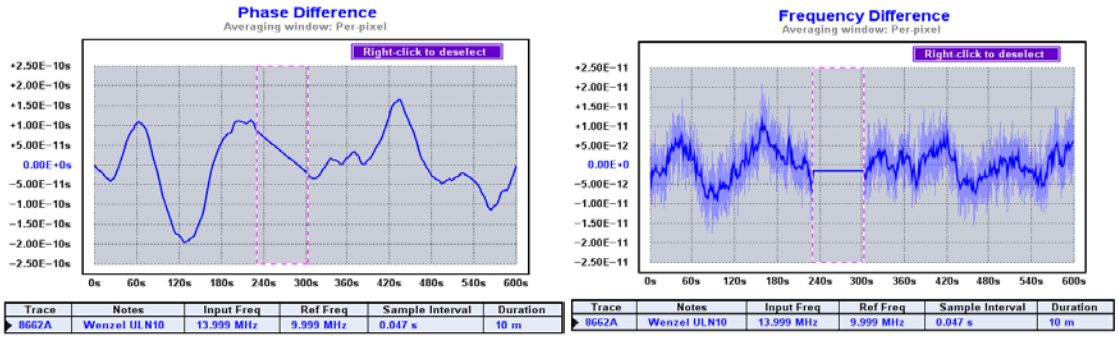
Trace properties (e)



As seen above, the **Edit→Trace properties (e)** dialog lets you change various properties associated with the selected plot that determine how its measurement data is presented. These properties were assigned their initial values in the acquisition dialog before the measurement began. Like the various instrument-specific dialogs on the **Acquire** menu, the trace-properties dialog may vary from the image shown above. The dialog has extensive "mouseover help" text that is maintained alongside TimeLab itself, so the individual fields will not be described here at length. Simply position your cursor over each field to learn more about it.

Some areas of the dialog may be disabled depending on the data type(s) available in the selected plot. Also, certain properties – notably those in the **Frequency Stability** area that determine the bin configuration for Allan deviation and other statistical measurements -- cannot be edited while acquisition is in progress. If you didn't enter the desired values in the acquisition dialog before data collection began, you'll need to wait for the measurement to finish before you can change them.

Flatten selected or zoomed phase data (Ctrl-f)



Above, the Flatten selected or zoomed phase data (Ctrl-f) command has replaced the selected region of the phase record with a straight line. The result appears as a flat spot in the Frequency Difference measurement view.

This command is supported only when viewing the selected plot's phase-record contents using **Measurement→Phase difference (p)** or **Measurement→Frequency-difference (f)**. In addition, a region within the plot must first be specified. You can specify a region with a Shift-Left drag operation that leaves the region boundaries visible within the existing graph, as in the images above, or you can simply drag with the left mouse button to zoom in on the desired region. In either case, the command will replace the data within the specified region of the selected plot's phase record with a straight line segment that connects the data points at the region boundaries. As seen in the figures above, a straight line in the phase record corresponds to a flat region in the **Measurement→Frequency difference (f)** view.

You can use **Edit→Flatten selected or zoomed phase data (Ctrl-f)** to remove glitches and outliers from an acquisition. Subsequently, **Edit→Undo last flatten or subtract operation (Ctrl-z)** can be used to restore the flattened data. Unless you absolutely must preserve the original length of the phase record, though, it's better to use **Edit→Remove selected or zoomed phase data (F4)** in most cases.

No active background tasks may be under way for the selected plot at the time this command is issued. Active background tasks are those in which the plot's memory-resident phase record is being written, such as during initial acquisition and when reloading a saved .TIM file. If the phase record has finished loading but the xDEV statistical traces are still being refreshed, the refresh operation will be cancelled, then restarted after the command finishes.

Remove selected or zoomed phase data (F4)

Like **Edit→flatten selected or zoomed phase data (Ctrl-f)**, this command is useful for removing glitches and outliers from the selected plot's phase record. Instead of replacing the affected region with a straight line, however, **Edit→Remove selected or zoomed phase data (F4)** shortens the phase record by removing the data altogether.

This command has the same requirements and constraints as **Edit→Flatten selected or zoomed phase data (Ctrl-f)**, except that it cannot be undone with **Edit→Undo last flatten or subtract operation (Ctrl-z).** Consider saving your original data to a .TIM file first so that it can be recovered if necessary.

When removing data from the middle of the phase record, you'll be asked if you want to maintain phase continuity. In virtually all circumstances you should respond "Yes" to this prompt. TimeLab will then ensure a seamless transition between the two remaining regions by applying a constant offset to the phase data following the removed region. Otherwise, an abrupt phase change due to lost continuity may appear as a glitch in the data.

No active background tasks may be under way for the selected plot at the time this command is issued. Active background tasks are those in which the plot's memory-resident phase record is being written, such as during initial acquisition and when reloading a saved .TIM file. If the phase record has finished loading but the xDEV statistical traces are still being refreshed, the refresh operation will be cancelled, then restarted after the command finishes.

Subtract global linear phase trend (frequency offset) (Ctrl-o)

Subtract global linear frequency trend (drift line) (Ctrl-l)

Subtract quadratic linear frequency trend (drift curve) (Ctrl-q)

These related commands are used to remove global offsets and trends from the selected plot's phase record. This is typically done to improve visibility of local trends and features whose magnitude is small compared to the Y-axis range. For detailed usage information and examples, as well as some alternative viewing methods that don't alter the phase record contents, see page 48.

Two commands on the Trace menu, **Phase/frequency traces begin at zero (z)** and **Show linear phase/frequency residual (r)**, provide non-destructive functionality similar to the **Edit→Subtract global linear phase trend (frequency offset) (Ctrl-o)** command. See page 77 for more information.

If the selected plot has any background tasks in progress when any of these commands are issued, an error message will be displayed. Background tasks are those in which the plot's memory-resident phase record is being accessed by a background thread during initial acquisition, when reloading a saved .TIM file, and when reconstructing the xDEV statistical traces after a previous modification to the phase record.

Undo last flatten or subtract operation (Ctrl-z)

This command rolls back the most recent change to the selected plot's phase record performed by any of the following other commands:

- **Edit→Flatten selected or zoomed phase data (Ctrl-f)**
- **Edit→Subtract global linear phase trend (frequency offset) (Ctrl-o)**
- **Edit→Subtract global linear frequency trend (drift line) (Ctrl-l)**
- **Edit→Subtract quadratic linear frequency trend (drift curve) (Ctrl-q)**
- **Edit→Trace properties (e) (applies to Rescale Phase only)**

A second "undo" command will redo the last-reverted action.

If the selected plot has any background tasks in progress when this command is issued, an error message will be displayed. Background tasks are those in which the plot's memory-resident phase record is being accessed by a background thread during initial acquisition, when reloading a saved .TIM file, and when reconstructing the xDEV statistical traces after a previous modification to the phase record.

Trace Menu

The **Trace** menu contains options that affect the way measurement data is processed and rendered as traces on the graph.

Unlike the **Edit** menu options, the **Trace** options apply to all visible traces, rather than only the selected plot, and they never alter the underlying measurement data. Most options on the **Trace** menu apply only to a given measurement family – xDEV statistics, phase/frequency differences, or phase noise/AM noise measurements. For example, **Trace→Averaging window for phase/frequency traces** has no effect on phase noise or Allan deviation traces.

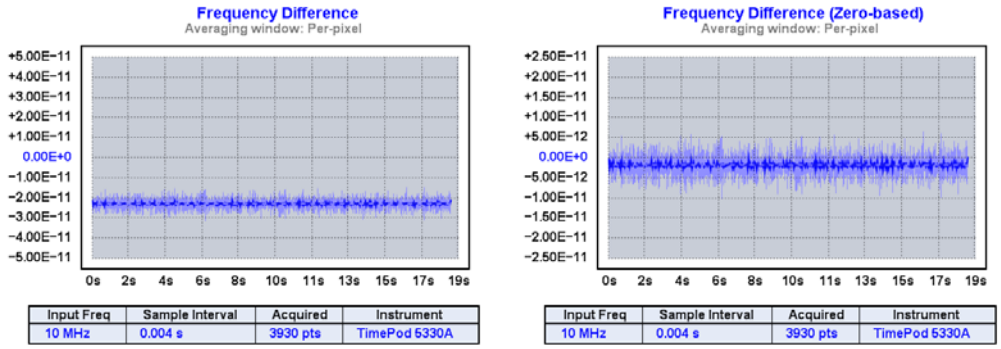
The majority of **Trace** options are simple on/off toggles. In the **Measurement→Phase difference (p)** display, for instance, applying the **Edit→Subtract global linear phase trend (frequency offset) (Ctrl-o)** command to each visible trace would have the same visible effect as **Trace→Show linear phase/frequency residual (r)**, as discussed below. But **Trace→Show linear phase/frequency residual (r)** does not actually subtract the trend from the phase record. Issuing the command again will restore normal trace-rendering behavior.

Phase/frequency traces begin at zero (z)

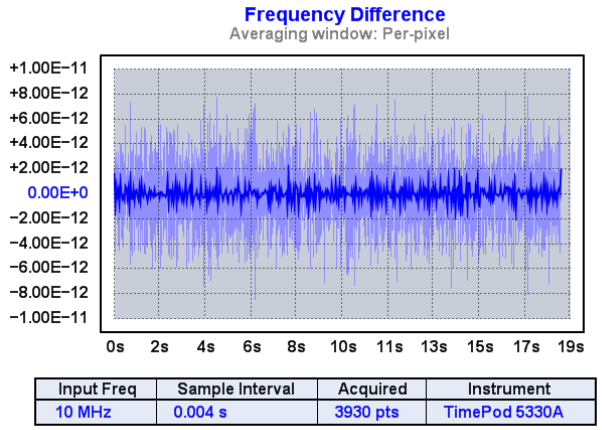
Show linear phase/frequency residual (r)

These two commands can be used individually or together to maximize visible detail in phase/frequency plots.

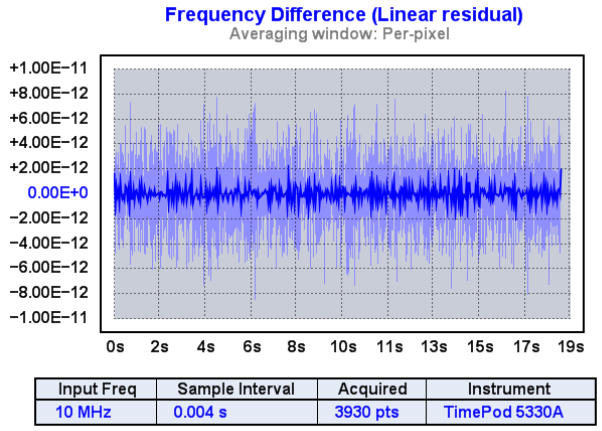
When enabled, **Trace→Phase/frequency traces begin at zero (z)** causes all phase- and frequency-difference traces to be displayed with their first data point pinned to Y=0.0. In unzoomed views – or in zoomed views with **Trace→Phase/frequency Y axis unlocked in zoom mode (y)** disabled to permit automatic Y-axis scaling – this will force the left end of the trace to the midpoint of the graph's Y axis. Since autoscaled Y axes in TimeLab are symmetrical about zero, this may allow the trace to occupy more of the visible graph area:



When used in the **Measurement→Frequency difference (f)** view, the effect of **Trace→Phase/frequency traces begin at zero (z)** is somewhat similar to applying **Edit→Subtract global linear phase trend (frequency offset) (Ctrl-o)** to each visible plot. It's not *quite* the same, though, because a trace's first data point is not necessarily anywhere near its trend line. Subtracting the trend will often do a better job at centering individual traces symmetrically about the Y axis, as seen below.



Of course, **Edit→Subtract global linear phase trend (frequency offset) (Ctrl-o)** has the drawback of altering the selected plot's phase data record and removing potentially useful information (namely the phase trendline). Fortunately, there's also a non-destructive global option that has the same visual effect, **Trace→Show linear phase/frequency residual (r)**:



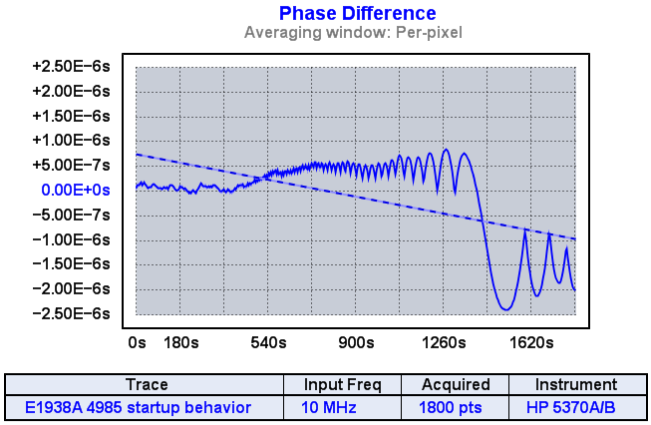
These two commands, **Show linear phase/frequency residual (r)** and **Phase/frequency traces begin at zero (z)**, are among the most commonly-used options on the **Trace** menu. If you spend much time working with phase- and frequency-difference traces, their r and z keyboard shortcuts will quickly become familiar. z can increase visible

detail while preserving the graph's original shape, while r is better at revealing short-term noise and other effects that become harder to spot as the graph expands to accommodate a larger overall trend.

Some additional usage examples can be found on page 48.

Show linear phase/frequency trend (Ctrl-t)

When enabled, this option causes TimeLab to display a dashed line on unzoomed phase- and frequency-difference traces to indicate the data's linear trend.



Trace	Input Freq	Acquired	Instrument
E1938A 4985 startup behavior	10 MHz	1800 pts	HP 5370A/B

The trend line is not displayed in magnified views, or when **Trace→Show linear phase/frequency residual (r)** is enabled.

Phase/frequency Y axis unlocked in zoom mode (y)

By default, the Y axis in phase- and frequency-difference measurement graphs is automatically scaled to accommodate the largest absolute value in all visible traces. The peak magnitude is rounded up to the next decade, half decade, or quarter decade, then mirrored to place zero at center scale. This approach allows the 10-division Y axis to be labeled in "nice" multiples of 1, 2, or 5.

When zooming in with the mouse to magnify a portion of the visible trace(s), it's sometimes helpful to toggle the Y-autoscaling algorithm off by selecting **Phase/frequency Y axis unlocked in zoom mode (y)**. This will have no effect on an unzoomed display, but when dragging with the left mouse button to magnify a desired area of the graph, you'll be able to move in the both X *and* Y directions, instead of only being able to drag a pair of vertical cursors. When you release the button, the resulting zoomed view will conform to the specified extents in both directions.

With the Y axis unlocked, the **Display→Y zoom in (})** and **Display→Y zoom out ({)** commands will be usable, and the scroll wheel will expand or contract the magnified region in both directions at once. Dragging with the middle mouse button will scrub the zoomed region in both the X and Y directions.

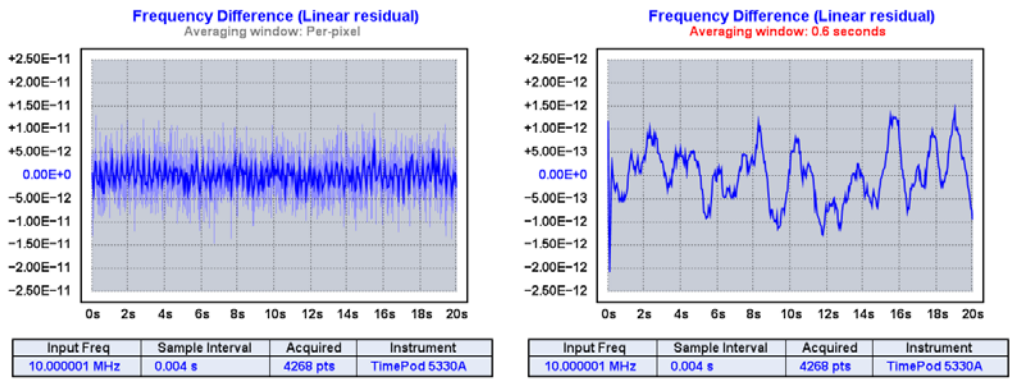
You can enable Y-axis autoscaling in zoom mode at any time. A quick way to "zoom out" in Y while maintaining the current X position and magnification factor is to tap the y key twice in succession – once to temporarily re-enable autoscaling, and again to restore 2D navigation with the middle button, scroll wheel, or ( ) { } shortcuts.

Averaging window for phase/frequency traces (g)

Increase averaging window (Ctrl +)

Decrease averaging window (Ctrl -)

These commands allow you to specify the length of an *averaging window* in seconds that will be applied to phase- and frequency-difference traces. The averaging process makes it easier to spot trends and patterns that may be obscured by short-term noise. Setting the window length to its default value of zero selects a hybrid "per-pixel" peak/average detection algorithm (discussed below.)



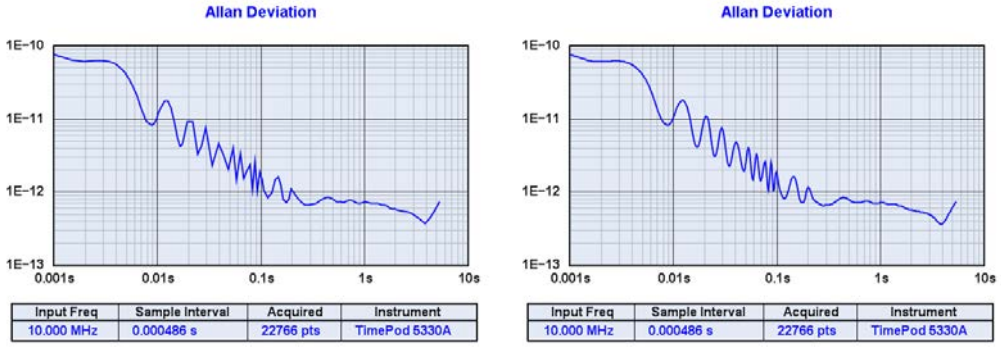
Warning: Averaging introduces time lag which may cause unexpected results with some commands and display options. It's recommend that averaging be disabled before selecting regions of the phase record for any operations, or when evaluating absolute phase differences or frequency differences. At long averaging times, a substantial baseline offset can exist between the beginning and end of the trace, since there's less historical data to contribute to the average near the beginning of the record. The trace may appear to "fall off a cliff," where noisy initial data gives way to a relatively-featureless trace as the averaging process takes effect. This effect is just beginning to appear in the 0.6s-averaged plot above.

As a reminder that averaging is in effect, nonzero averaging times will be displayed in red above phase- and frequency-difference graphs. You can quickly disable averaging with the keyboard sequence g 0 <Enter>.

Averaging also determines how phase records are decimated for display in the phase- and frequency difference views. When rendering a trace whose phase record length in samples exceeds the graph width in pixels, the samples that fall within each graph column must either be peak-detected or combined by averaging. When the averaging window is set to zero seconds, TimeLab effectively does both, using a "per-pixel" algorithm to provide as much information about the skipped phase-record samples as possible. As shown in the first plot above, a shaded vertical line is drawn in each pixel column that connects the minimum and maximum data values that fall within that column. The darker portion of the trace in the middle of the shaded area represents the average of *all* data points covered by each individual column. The effective averaging time is not fixed, but is equal to the duration of the visible portion of the trace divided by the width of the graph in pixels.

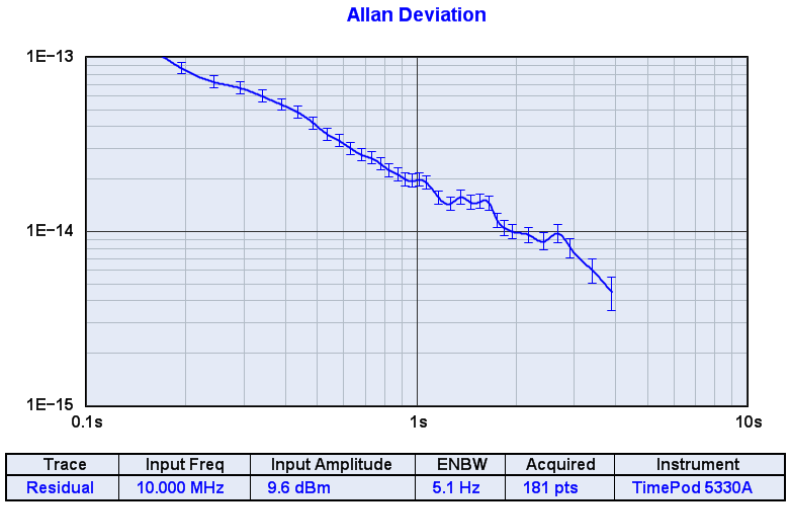
This algorithm is simple, fast, and effective, but since the peak-detected line segments are drawn in a lighter shade of the trace color, it's easy to overlook brief glitches that fit within a single pixel column. Consider enabling **Trace→Tick marks (k)** when looking for outliers in per-pixel mode.

Draw xDEV traces with spline interpolation (i)



By default, screenspace points corresponding to the tau bins in Allan deviation and related statistical views are connected with a cubic spline. Spline interpolation results in a smooth, visually-appealing curve, but it can distort or exaggerate ringing artifacts and other abrupt transitions in the graph. When **Trace→Draw xDEV traces with spline interpolation (i)** is toggled off, straight lines are used to connect the xDEV bins (above left).

Show xDEV error bars (Ctrl-e)


Allan Deviation
1E−13
1E−14
1E−15
0.1s
1s
10s
Trace | Input Freq | Input Amplitude | ENBW | Acquired | Instrument
Residual | 10.000 MHz | 9.6 dBm | 5.1 Hz | 181 pts | TimePod 5330A

When enabled, this option draws “error bars” at each tau bin in Allan deviation and other statistical plots. These bars approximate the ± one-sigma (68%) confidence interval at ± σ/sqrt(N/M), where N is the number of phase-record samples that contribute to each bin and M is the bin’s τ_0 multiple.

This calculation does not take either the deviation type or the noise slope into account, so it should be considered a rough estimate of the actual confidence interval obtainable through more sophisticated offline analysis. Consider exporting your phase data to Stable32 (page 66) if your application requires better confidence-interval estimates.

Clip xDEV traces by noise bandwidth (Ctrl-b)

This display option is used to suppress artifacts near the tau-zero points in Allan deviation and other statistical plots that can appear when incoming data is oversampled for lowpass filtering. xDEV values at tau intervals close to the sample rate may appear to "droop" in such cases, since the data has been band-limited at a cutoff frequency below its Nyquist rate.

A discussion of this effect appears on page 35.

Clip xDEV traces by confidence (Ctrl-v)

This display option is used to avoid rendering Allan deviation and other xDEV traces beyond the point where insufficient data is available for good statistical confidence. It is enabled by default.

Specifically, when this option is enabled, the only xDEV bins that are drawn are those whose phase data sample count (N) is >= 2x their τ_0 multiple (M).

Show correlation gain for selected noise trace (Ctrl-g)

Intended primarily as a diagnostic aid, this option is supported when viewing phase noise or AM noise plots from instruments such as the TimePod 5330A that support real-time cross correlation. When enabled, the FFT segments in the selected phase noise or AM noise plot will be color-coded for identification. A table will appear as an overlay in the graph area that displays the number of cross-spectrum averages performed so far in each color-coded segment, along with the theoretical improvement in ADC noise expressed in dB.

The estimated improvement in each segment's effective noise floor is calculated as $10 * \log_{10}(\text{sqrt}(N))$ dB, where N is the segment's number of averages.

Trace smoothing is not performed on the selected noise plot when this option is enabled.

Show FFT segment filter slopes (Ctrl-i)

This option is a diagnostic aid intended for use when editing FFT segment table files for phase noise and AM noise acquisition with TimePod hardware.

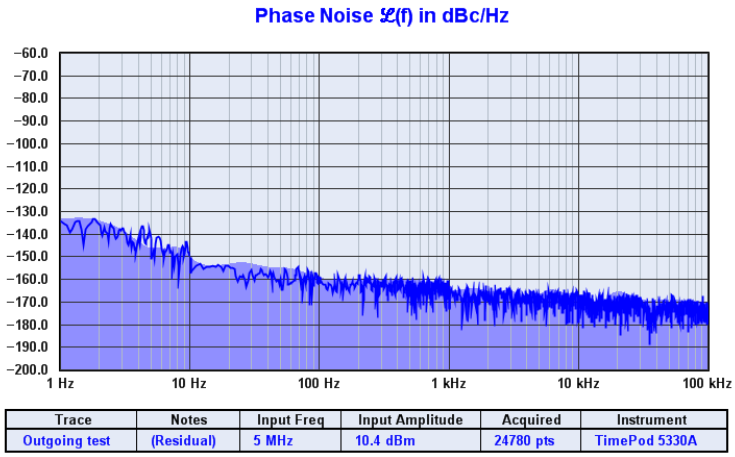
When acquiring noise plots on the 5330A, data for each visible FFT segment undergoes highpass and/or lowpass filtering based on the declarations in the *segment table*, accessible via the **Advanced** tab in the TimePod acquisition dialog. By rendering the entire extent of each segment record, **Trace→Show FFT segment filter slopes (Ctrl-i)** provides a visual display of the filter skirts for each segment, as well as the overlap between them. In this mode, you can temporarily hide the skirts by holding the SHIFT key.

For detailed information about the segment table's format and contents, refer to the comments within the default segment table file. This file can be accessed by pressing the **Advanced→Edit segment table** button in the TimePod acquisition dialog.

Most users will not need to edit the segment table.

Show imaginary part of cross spectrum (Ctrl-F3)

Show estimated instrument noise (F2)



Phase noise and AM noise plots acquired with the TimePod 5330A and Symmetricom 512X analyzers can be displayed with a shaded area that provides an estimate of the instrument noise floor, if you enable **Trace→Show estimated instrument noise (F2)**.

In the 5330A's case, the floor estimate is simply a heavily-smoothed rendition of the imaginary portion of the cross-spectrum average shown by the **Show imaginary part of cross spectrum (Ctrl-F3)** command. For more information about the imaginary cross spectrum and the Ctrl-F3 command in particular, see "Understanding instrument spurs" on page 57.

Use caution when relying on the noise-floor estimate. While reasonably accurate, it may appear artificially high in the vicinity of instrument spurs. Conversely, thermal effects can raise the real measurement floor slightly above the shaded floor estimate. As with ADEV measurements, the best way to obtain "proof of performance" for analyzers in this class is to run residual tests at similar frequencies and signal levels.

Particularly when viewing multiple traces in **Display→Overlay (o)** mode, it may be preferable to leave the floor display turned off in order to reduce display clutter.

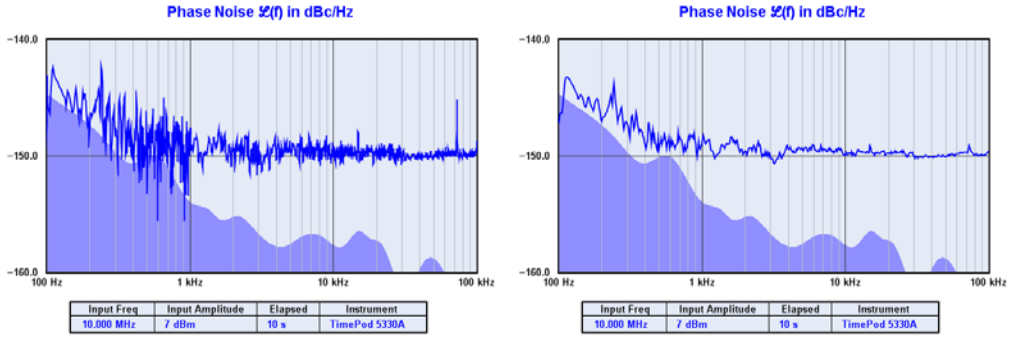
Note that when **Show imaginary part of cross spectrum (Ctrl-F3)** is active, the **File→Export AM/PM noise trace** command will export the imaginary cross spectrum trace data rather than the normal AM noise or phase noise trace.

Mark spurs in noise traces (Ctrl-m)

Suppress spurs in noise traces (Ctrl-s)

For a detailed discussion of these commands and other aspects of spur detection and rendering in phase noise and AM noise plots, see page 55.

Smooth noise traces (Ctrl-w)



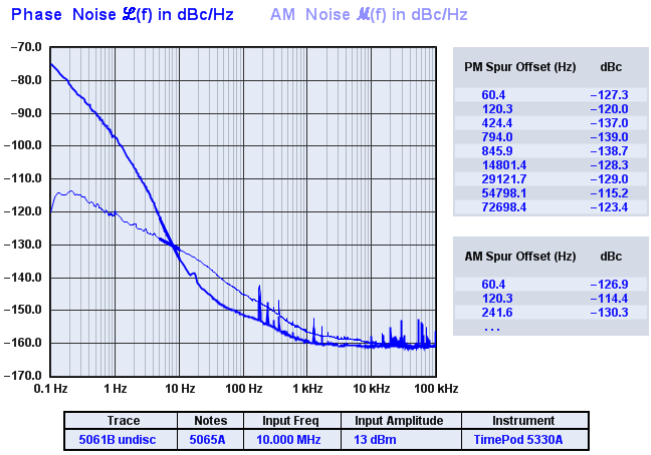
This command applies some light exponential smoothing to traces in the phase noise and AM noise measurement views. Detected spurs are always removed from smoothed traces. If enabled, smoothing is turned off automatically when the **Trace→Suppress spurs in noise traces (Ctrl-s)** command is used to toggle spur attenuation.

To obtain a smoother trace with cross-correlating analyzers such as the TimePod 5330A and Symmetricom 512x models, it's often better to let the measurement run longer. As the trace converges on the true noise level its variance will diminish, resulting in a more accurate measurement with less visible "grass."

Show raw PN channel trace(s) (Ctrl-r)

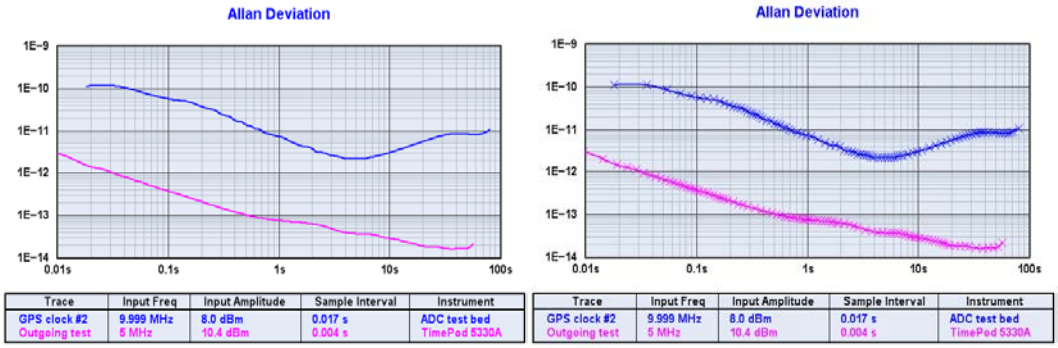
This command is intended for diagnostic purposes only. It displays the raw phase noise spectrum from the ADCs in both measurement channels of the 5330A.

Show AM noise in PN view (F8)



When enabled, this command renders a copy of the AM noise trace in the **Measurement→Phase noise (P)** view, using a lighter color or trace weight. Both PM and AM spur tables are displayed, space permitting.

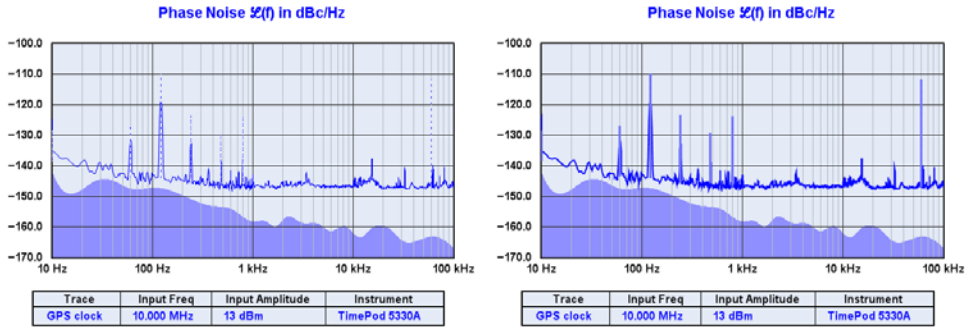
Tick marks (k)



This command toggles tick marks on and off.

Tick marks are supported in most trace types and display modes. They can be especially handy when examining phase/frequency difference plots. In cases where many data points contribute to a single pixel column in the graph and no explicit averaging time has been specified, TimeLab's "per-pixel" renderer displays the average of all of each pixel column's contributing points, while the maximum and minimum points that fall within each pixel column are rendered in a lighter shade of the same color. The k command can help you spot brief glitches and frequency jumps in these plots that would otherwise go unnoticed.

Toggle trace thickness for current measurement (T)

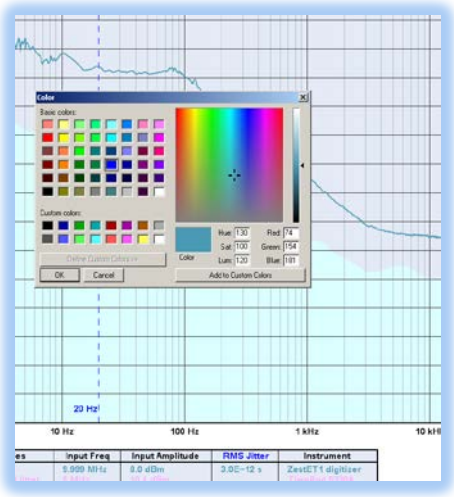


This command switches between heavy and light traces in the currently-active measurement view. TimeLab keeps track of the requested trace thickness for each **Measurement** menu entry.

Display Menu

Display-related options that aren't measurement-specific appear on the **Display** menu. Controls on this menu determine the visibility and selection status of loaded plots, the order in which plots appear in the legend table below the graph, and the choice of **Display→Overlay (o)** or **Display→Browse (b)** mode that determines whether all loaded plots are displayed or only the selected plot.

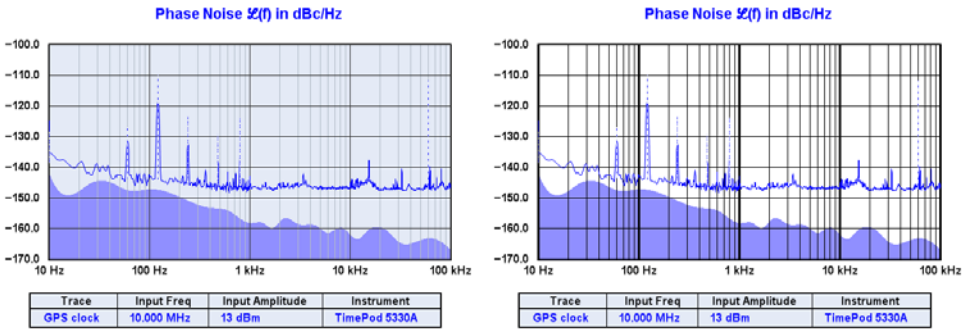
Color, contrast, font size, and graph-magnification options also appear on the **Display** menu, along with a choice of formats for the "mouseover" information at the upper left corner of the graticule.



Edit colors

This command leads to a second-level menu offering a choice of various graphical elements in the TimeLab window, from font and trace colors to table borders and backgrounds. Once an element is chosen, a standard Windows color palette dialog appears. As you click on the palette areas to change the selected color, the TimeLab window is updated in real time, making the color-selection process interactive and easy to use.

High contrast (C)



This command toggles between normal and high-contrast rendering. The latter option may be desirable when saving screenshots for publications or Web pages whose color fidelity may not be sufficient to reveal details in the plot such as minor graticule lines.

Numeric table (Ctrl-n)

The *numeric table* appears to the right of the graph area in each measurement view, unless toggled off with this command. The format of the table is different for each measurement type. In the Allan deviation and other statistical views, a chart displays the sigma(tau) values at assorted tau intervals. In the **Measurement→Frequency difference (f)** view, both drift/trend statistics and a high-precision frequency-count chart are displayed, while only the former is available in the **Measurement→Phase difference (p)** view. Finally, a table of detected spurs is displayed in the phase noise and AM noise views.

Show cursor time (Ss)

Show cursor time (Hh:Mm:Ss)

Show cursor time/datestamp

Do not show cursor values

These options determine the content of the "mouseover" cursor readout display in the **Measurement→Phase difference (p)** and **Measurement→Frequency difference (f)** views. When the mouse cursor is within the graph area in these views, the readout at upper left displays the time and Y-axis value corresponding to its position. You can choose to display the number of seconds relative to the beginning of the phase record (**Ss**), the same elapsed time value in **Hh:Mm:Ss** format, or the absolute **time/datestamp** when the data at the mouse cursor location was logged.

Note that **Display→Do not show cursor values** turns off the cursor readout in *all* views. To re-enable the cursor readout in the xDEV and noise displays, select any of the other three options.

The mouse cursor readout is not rendered when saving image files or capturing images with **File→Copy image to clipboard (Ctrl-c)**.

Browse plots one at a time (b)

Overlay all loaded plots (o)

Toggle visibility of selected plot (v)

Since TimeLab allows up to nine plots to be loaded and displayed at once, these three commands are needed to maintain a legible display. They are often used together in rapid succession, so they will be described together and referenced in abbreviated form.

Simply stated, in **b)rowse** mode only the selected plot is rendered. In **o)verlay** mode, all loaded plots that haven't had their **v)isibility** toggled off are rendered, with the graph scale adjusted as necessary to accommodate them.

It's common to switch between **b)rowse** and **o)verlay** mode when inspecting and comparing several plots, as the display can become rather crowded. At the same time, the **v)isibility** command is often needed when you want to view a subset of plots in o)verlay mode. **b)rowse** mode will exclude all but the selected plot from the display, while the **v)isibility** toggle gives you the control needed to display some, all, or none of the loaded plots in **o)verlay** mode. If you want to look at several overlaid plots while temporarily hiding one or two of them, the **v)isibility** toggle is the answer.

Select next plot in chart (+ or down arrow)

Select previous plot in chart (- or up arrow)

Move selected plot up (Ctrl-up arrow)

Move selected plot down (Ctrl-down arrow)

For an introduction to the "selected plot" concept and its related commands, refer to "Making your first measurements" on page 21.

The **Ctrl-up/down arrow** commands can be very helpful for organizing displays with multiple plots. Holding down the Ctrl key when you press an arrow key will not change the currently selected plot, but will instead move the selected plot up and down in the legend table. Since the trace colors in TimeLab are assigned based on each plot's position in the legend table, this feature can be useful if you don't like the color of a particular plot, or if a plot farther down the table is obscuring data in an earlier plot. (The **Ctrl-up/down arrow** commands are frequently handy when using **Trace→Show estimated instrument noise (F2)**, for instance.)

X zoom in ( ])

X zoom out ([ )

Y zoom in (})

Y zoom out ({)

The bracket and shift-bracket keys are useful for navigating zoomed phase- and frequency-difference traces on a laptop or other PC without a three-button scrolling mouse. These menu options can be thought of as placeholders for their respective keyboard shortcuts. Specifically, after a zoomed region has been selected by dragging with the left mouse button, these commands allow you to increase and decrease the "magnification factor" without returning to the unzoomed display.

On a desktop PC, a combination of scroll-wheel and middle-button input is normally the best way to navigate within phase/frequency traces. The scroll wheel expands and shrinks the magnification factor, while the middle mouse button allows you to "scrub" the zoomed trace data left and right.

These commands have no effect in zoomed phase noise or AM noise measurement views, or in Allan deviation or other statistical views. For more about TimeLab's zoom functionality, refer to "Navigating zoomed graphs" on page 49.

Decrease font size ((or Ctrl-mouse wheel)

Increase font size ( ) or Ctrl-mouse wheel)

Like the **X zoom/Y zoom** commands, the font-size control options are primarily meant as placeholders for their respective "hotkeys," the left and right parentheses. They provide easy access to a choice of several discrete font sizes. Also, as with most Web browsers and other newer Windows applications, scrolling with the mouse wheel while holding down the Ctrl key will expand or shrink the text within TimeLab's display window.

Legend Menu

The **Legend** menu determines which attributes are displayed for each visible plot in the legend table beneath the main graph. Some properties are flagged for display in the legend table by default, while others will not be displayed until you select them in the **Legend** menu.

Each TimeLab acquisition driver can contribute its own entries to the **Legend** menu based on the property values in the plots that it generates. When a plot is acquired or loaded from a .TIM file, TimeLab checks for the presence of any displayable properties that it hasn't yet encountered. Any unfamiliar properties are logged in **TIMELAB.INI** and added to the **Legend** menu.

As a result, the content of the **Legend** menu will vary – often greatly – from one TimeLab installation to the next. (Note that after you select a property for display in the legend table, its value will be blank for all plots that don't include that particular property.)

As with other data stored in **TIMELAB.INI**, reinstalling TimeLab does not erase the list of known **Legend** menu entries. These entries can be cleared only by manually editing or deleting **TIMELAB.INI**, or by using **File→Reset all parameters, options, and settings at next startup**. Refer to the latter command for more information about **TIMELAB.INI**.

The list of currently-defined .TIM file properties appears below. Many of these options are 'private' to a specific hardware driver and are not eligible for display on the legend table. They are listed here for reference only. A few of the properties are defined by TimeLab itself and will appear on the **Legend** menu by default in all installations.

ADC Clock	Nominal acquisition clock rate in Hz
Add'l AM Gain	Optional user-specified gain value added to AM noise traces
Add'l PN Gain	Optional user-specified gain value added to phase noise traces
Address	Address of instrument used to acquire data
Attenuate Instrument Spurs	True if known instrument spurs are suppressed automatically
Auto DUT	True if the DUT frequency matches the frequency at the analyzer input jack
Auto Input	True if input frequency measured automatically
Auto Ref	True if external reference frequency measured automatically
Automatic Configuration	True if the TimeLab driver should infer various measurement params based on incoming data
Autosense Rate	True if the TimeLab driver should automatically measure the sample interval
Beatnote	Amplitude in dBm of calibration beatnote
Bin Density	Determines the number and distribution of tau points in statistical deviation measurements
Bin Threshold	Minimum number of data points required to display the statistical deviation at a given tau
BW Index	Index or other identifier for a given measurement bandwidth
Cancel CM Drift	True if common-mode drift (e.g., due to ADC clock warmup) is corrected in software
Ch 0	Combination of physical ADC channel(s) contributing data to FFT channel 0
Ch 1	Combination of physical ADC channel(s) contributing data to FFT channel 1
Channel	Identifies the channel number associated with a multichannel acquisition
Counter Connection Type	Interface used for acquisition from TIC or frequency counter
Data Type	Type of the incoming data (e.g., phase, frequency differences, or absolute frequency)
Decimation	Hardware decimation ratio
Device ID	Instrument serial number or other identifier
Driver	Identifies the TimeLab .TLL driver used for acquisition
Duration	Measurement duration
Duration Type	Measurement duration type
DUT Freq	Frequency of DUT provided by user, prior to any conversion, multiplication, or division
EFC	Acquisition clock EFC DAC value
ENBW	Equivalent noise bandwidth in Hz of recorded data
ENBW Factor	Equivalent noise bandwidth of recorded data, expressed as a fraction of the data rate
EOS Character	End-of-string character for data source
FFT Win #	Index or other identifier for FFT window function used for noise measurement
FFT Window	FFT window function used for noise measurement
Field #	Numeric field position from which data was extracted
FPGA Bitfile	Name of the .bit file used to configure the FPGA prior to acquisition
FPGA Version	FPGA firmware version
Frequency	Incoming data consists of absolute frequency readings
Frequency Difference	Incoming data consists of fractional frequency differences
FRQ Max Offset	Maximum offset frequency for spectrum display
FRQ Max Scale	dBm value at top of spectrum display
FRQ Min Offset	Minimum offset frequency for spectrum display
FRQ Min Scale	dBm value at bottom of spectrum display
HP 5313xA Mode	True if the TimeLab driver should ignore statistics data from an HP 53131A/53132A counter

Imported From	Name of file or device from which data was imported
Incoming Sample Interval	Time in seconds between successive data points from the instrument
Input Amplitude	Approximate input amplitude in dBm
Input Estimated	True if input frequency is a low-precision estimate that should be rounded for display
Input Freq	Input frequency
Input Splitter	True if input amplitudes should be adjusted by 3 dB for internal or external RF splitter
Instrument	Instrument model used to acquire data
Interface	Configuration details for the GPIB adapter, serial or LAN adapter, or other interface
IP Address	IP address or hostname of instrument used to acquire data
L(f) Factor	Additional value used to convert dBc/Hz readings into L(f) SSB phase noise
L(f) Max Scale	dBc/Hz value at top of SSB noise graph
L(f) Min Scale	dBc/Hz value at bottom of SSB noise graph
LNA dB	Gain provided by LNA stage
Measure AM Noise	True if measurement contains AM noise data
Measure Frequency Spectrum	True if measurement contains two-sided frequency spectrum data
Measure PM Noise	True if measurement contains phase noise data
Measure Stability	True if measurement contains phase/frequency stability data
MJD	Modified Julian Date when the measurement was triggered
Multichannel input amplitude	Approximate input amplitude in dBm for each channel in a multichannel measurement
New t0	New tau-zero interval for resampled phase data
Notes	Additional measurement notes, specified at acquisition time or via Edit->Trace Properties
Options	Option codes associated with instrument
Output Decimation	Final decimation ratio for output samples
Output Sample Rate	Rate at which data samples are generated by the TimeLab driver
Overlap %	Overlap between successive FFT buffers
Overlapped	True if overlapped FFT processing is enabled for faster measurement
Phase	Incoming data consists of phase-difference values
PN Max Offset	Maximum offset for SSB noise graph (which will be rounded up to the nearest decade)
PN Min Offset	Minimum offset for SSB noise graph (which will be rounded down to the nearest decade)
Port	IP port of instrument used to acquire data
Precision	Precision of incoming floating-point data (e.g., 0=single, 1=double)
Prologix Compatibility Mode	True if serial ports with FTDI D2XX drivers should be treated as Prologix GPIB adapters
Property Page	Selected property page number
Read Existing Data	True if existing text from the file/device should be skipped prior to acquiring new data
Ref Amplitude	Approximate reference amplitude in dBm
Ref Ch	Physical channel(s) acting as reference
Ref Estimated	True if reference frequency is a low-precision estimate that should be rounded for display
Ref Freq	Reference frequency
Rescale Factor	User-supplied rescale factor for phase data
Rev	Detailed instrument information, including firmware version and/or options
RS-232 Setup	Configuration string for RS-232 COM port, if used
Sample Interval	Time in seconds between successive data points
Sample Rate	Incoming data rate from instrument in samples per second
Scale Factor	Numeric scale factor used to convert TI readings to seconds or frequency readings to Hz
Scale Jitter to DUT	True if jitter and carrier/noise values should be based on DUT rather than input frequency
Scale PN to DUT	True if phase noise values should be increased by 20 * log10 (DUT freq / input freq)
Segment Table	User-specified tag field from segment table used for noise acquisition
Specified Input Frequency	Input frequency in Hz provided by user
Specified Reference Frequency	Reference frequency in Hz provided by user
Spur Threshold	Minimum amplitude relative to local average at which a line will be recognized as a spur
Stability Ch	Input channel(s) for stability measurement
Stability Channel Count	Number of stability-measurement channels
Stop Condition	Termination condition for measurement
Time/Date	Time and date when the acquisition file was created
Timestamp	Incoming data consists of absolute timestamps in seconds
Trace	Trace caption text, as specified at acquisition time or edited with Edit->Trace Properties
Trace History	Number of historical xDEV traces displayed
TSC Phase Data Rate	Phase data rate from TSC 512X-compatible timing analyzer
Unwrapped	True if phase data has already been unwrapped by the instrument driver
Unwrapped Phase	Incoming data consists of monotonic phase differences that do not contain phase wraps
USB Version	USB firmware version
Use Input Rate	True if driver's data rate matches the incoming data rate from the instrument
Warnings Treated as Errors	True if warnings during data acquisition should terminate the measurement
Wrap Period	Wraparound period in seconds for incoming timestamps

Measurement Menu

The **Measurement** menu selection determines how TimeLab displays the data in any plots that have been acquired or loaded. One measurement type may be selected at a time.

As discussed on page 27, TimeLab plots (and their associated .TIM files) use independent data records for different measurement classes. Some instruments like the TimePod 5330A can acquire data for all measurements supported by TimeLab, but most are limited to specific measurement types. For instance, acquisitions made with a frequency counter or TIC may be viewed with the **Allan deviation (**a**)**, **Modified Allan deviation (**m**)**, **Hadamard deviation (**h**)**, **Time deviation (**t**)**, **Phase difference (**p**),** or **Frequency difference (**f**)** selections on the menu, but not **Phase noise (**P**)** or **AM noise (**A**).**

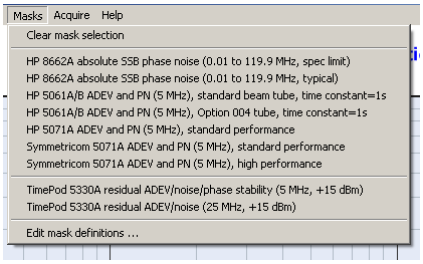
Regardless of the instrument used, all supported measurement records are acquired and processed simultaneously by TimeLab. After launching a 5330A acquisition with all of the **Available Measurements** options selected in the acquisition dialog, for example, you can switch between all of the entries in the **Measurements** menu to view your data as it arrives.

Plots that don't include the necessary data for the selected measurement type will not be rendered. When the TimeLab window is empty despite the presence of one or more loaded or acquired plots, a brief explanatory message will be displayed in the color associated with each plot to indicate why nothing is being drawn.

See "Making Measurements" on page 29 for a detailed discussion of the measurement types supported by TimeLab.

Masks Menu

The **Masks** menu is almost entirely user-configurable. It consists of a list of *mask definitions* for optional use in TimeLab. Masks are useful in production test applications where fast, reliable, and repeatable pass/fail judgments are required. You can also use mask testing for proof-of-performance verification of the TimePod and other instruments supported by TimeLab.


Masks
Acquire
Help
Clear mask selection
HP 8662A absolute SSB phase noise (0.01 to 119.9 MHz, spec limit)
HP 8662A absolute SSB phase noise (0.01 to 119.9 MHz, typical)
HP 5061A/B ADEV and PN (5 MHz), standard beam tube, time constant=1s
HP 5061A/B ADEV and PN (5 MHz), Option 004 tube, time constant=1s
HP 5071A ADEV and PN (5 MHz), standard performance
Symmetricom 5071A ADEV and PN (5 MHz), standard performance
Symmetricom 5071A ADEV and PN (5 MHz), high performance
TimePod 5330A residual ADEV/noise/phase stability (5 MHz, +15 dBm)
TimePod 5330A residual ADEV/noise (25 MHz, +15 dBm)
Edit mask definitions ...

At startup, Timelab constructs the **Masks** menu based on the contents of the user-editable file **masks.txt**. By default, **masks.txt** is stored in the **%ALLUSERSPROFILE%\Documents\TimeLab\Masks** folder. This folder's location will vary from one Windows version to the next. For example, in a typical Windows 7 installation on drive C: it can be found at **C:\ProgramData\Documents\TimeLab\Masks**, while under Windows XP SP2 the directory is located at **C:\Documents and Settings\All Users\Documents\TimeLab\Masks**.

Generally, the easiest way to read and modify **masks.txt** is by selecting the last entry on the **Masks** menu, **Mask→Edit mask definitions**. This option will open **masks.txt** in Windows Notepad. You can then create your own mask definitions, edit existing masks, remove masks from the menu, or simply review the mask-testing features and capabilities in the current version of TimeLab.

Important: To preserve any user-created mask definitions, your existing **masks.txt** file will not be overwritten when a new copy of TimeLab is installed. The only exceptions to this rule occur when new features are introduced that render the mask file format incompatible with newer versions of TimeLab. In such cases, the TimeLab setup program will make a backup copy of **masks.txt** in the same directory, under the name **masks_backup.txt**. You will then need to transfer any user-created mask definitions from **masks_backup.txt** to the new **masks.txt** file.

Clear mask selection

The first command on the **Masks** menu simply deselects any currently-selected mask, removing its limit line(s) from the applicable measurement views.

User-defined mask entries

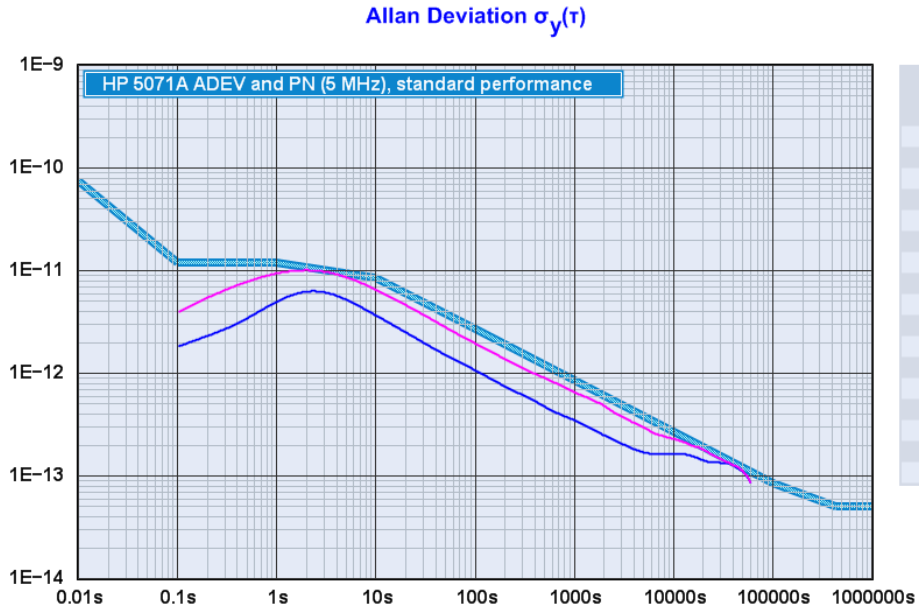
Menu entries appearing between **Mask→Edit mask definitions** and **Mask→Edit mask definitions** are defined in **masks.txt**. Only one mask may be selected at a time. Once selected, the mask's title and limit line will be visible only in measurement views corresponding to valid limit lines in the mask definition.

To see the results of the selected mask test, you'll need to select one of these views from the **Measurement** menu *and* enable the **Legend→Mask Result**, **Legend→Mask Margin**, or both. Pass/fail results and margins will then appear in the legend table below the graph, updated in real time as the measurement progresses.

For detailed instructions and hints, carefully review the comments in **masks.txt**. The latest usage guidelines for the current TimeLab release will appear as comments in this file.

Edit mask definitions . . .

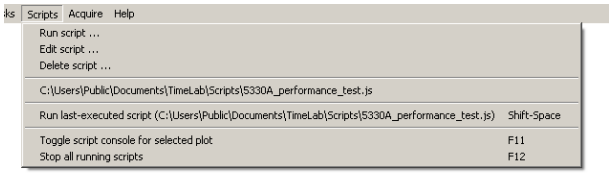
As noted above, the last command on the **Masks** menu opens **masks.txt** in Notepad for editing and review.



Tau	Sigma(Tau)
1s	5.13E−12
2s	6.37E−12
4s	5.74E−12
8s	4.14E−12
10s	3.67E−12
20s	2.51E−12
40s	1.70E−12
80s	1.20E−12
100s	1.07E−12
200s	7.48E−13
400s	5.37E−13
800s	3.83E−13
1000s	3.51E−13
2000s	2.50E−13
4000s	1.84E−13
8000s	1.66E−13
10000s	1.67E−13

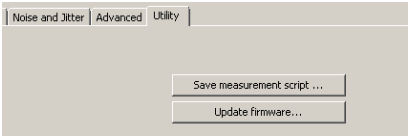
Trace	Notes	Input Freq	Mask Result	Mask Margin	Imported From
HP 5071A	HP 5065A	10 MHz	Pass	6.76E-015	K:\Program Files\Miles Design\TimeLab\5071A.txt
HP 5061B	HP 5065A	10 MHz	Pass	5.73E-015	K:\Program Files\Miles Design\TimeLab\5061B.txt

Scripts Menu



TimeLab includes an embedded JavaScript engine[13] that can run automated test scripts using the TimePod 5330A. (Script-based access to other instruments such as time-interval counters is not currently supported.) Numerous API functions are provided to support a variety of applications, ranging from the standard TimePod performance verification procedures included with TimeLab to user-developed test scripts. The **Scripts** menu helps you edit, run, and otherwise manage your collection of test scripts.

Creating a new test script



The easiest way to create a new test script is to set up your measurement in the TimePod acquisition dialog just as you normally would. Then, rather than selecting **Start Measurement**, click the **Save measurement script** button on the **Utility** tab. The Windows file dialog that appears will prompt you to save the current set of measurement parameters to a JavaScript (.js) file. When executed later with **Scripts→Run script**, the script will launch a measurement with the same acquisition parameters and options.

As an alternative, you can use **Scripts→Edit script** to create modified copies of the scripts that are included with TimeLab, such as **5330A_performance_test.js**, **5330A_ADC_test.js**, and **OCXO_2hr_stability_test.js.** These programs are more elaborate than the elementary scripts created by **Save measurement script**. They exercise virtually all of the capabilities of the TimeLab script API as well as those of the TimePod 5330A itself.

By default, script files are stored in the **%ALLUSERSPROFILE%\Documents\TimeLab\Scripts** folder. This folder's location will vary from one Windows version to the next. For example, in a typical Windows 7 installation on drive C: it can be found at **C:\ProgramData\Documents\TimeLab\Scripts**, while under Windows XP SP2 the script directory is located at **C:\Documents and Settings\All Users\Documents\TimeLab\Scripts**.

For a detailed listing of script API functions, refer to the JavaScript API Function Reference section on page 115.

[13] http://code.google.com/p/v8/

Run script …

To launch a test script, select **Scripts→Run script . . .** and choose the desired script's JavaScript file (.js) in the file dialog that appears. TimeLab will load the script into memory and execute its "global scope" to initialize any variables declared outside of functions or event handlers. Control then passes to the script's mandatory **EventRunScript()** function.

Once launched, the script will continue to run until one of the following conditions is true:

- A script exception or other fatal error occurs

- The **ScriptEnd()** API function is called by the script prior to returning from an event handler

- The script returns from an event handler function such as **EventRunScript()** or **EventAcqDone()** with no plots associated with it, either because the user closed the plot(s) manually or because the event handler did not launch any acquisitions. (However, if the script has requested timer service with **TimeSetTimer()**, it will be allowed to continue running.)

- The **Scripts→Stop all running scripts (F12)** command is issued by the user

- The user exits from TimeLab

Any errors encountered during either loading or execution will be reported in the status line at the bottom of the TimeLab window.

Edit script …

This command simply launches an instance of Notepad on the specified .js file, allowing you to read, modify, and/or copy the script.

Delete script …

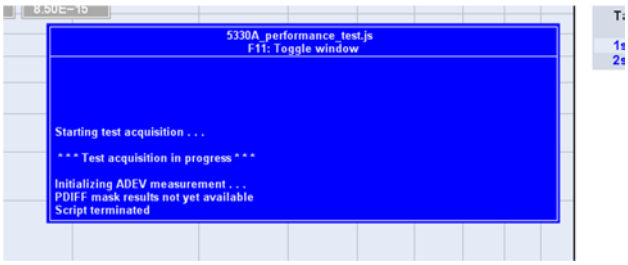
To delete a test script, select **Scripts→Delete script . . .** and specify the .js file to delete. The .js file will be deleted without further prompting.

Run last-executed script (Shift-Space)

Using Shift-Space to re-run the most recently executed script is analogous to pressing the Space bar by itself to bring up the most recently-used acquisition dialog. The script will be relaunched without further prompting from TimeLab.

Toggle script console for selected plot (F11)

When a measurement is started under script control, TimeLab will display an overlay containing any error or status messages issued by the script. This console overlay is shown by default whenever the measurement is otherwise visible on the graph.


5330A_performance_test.js
F11: Toggle window
Starting test acquisition . . .
* * * Test acquisition in progress * * *
Initializing ADEV measurement . . .
PDIFF mask results not yet available
Script terminated

As the script console overlay tends to obscure the data being plotted, the F11 key provides a convenient way to hide or show the overlay for the currently-selected plot.

Stop all running scripts (F12)

Stops any scripts that are currently running, as well as any measurements associated with those script(s).

Acquire Menu

TimeLab's principal mission is to serve as the user interface for the TimePod® 5330A Programmable Cross Spectrum Analyzer from Miles Design, but support for many other instruments is also provided. The **Acquire** menu includes a list of all instruments recognized by TimeLab, followed by a small number of commands that manage acquisition operations.

When selected, each entry on the **Acquire** menu that corresponds to a given instrument will bring up an *acquisition dialog.* Acquisition dialogs are used to configure the instrument and initiate data collection. Every acquisition in TimeLab runs in its own thread, so you can load or acquire other plots while data collection is in progress. Acquired data can be saved at any time, even during the measurement; likewise, all options on the **Trace** and **Display** menus apply equally to existing plots and those that are currently being acquired.

While acquisition dialogs are straightforward to work with, their layout and functionality varies greatly from one instrument to the next. Each dialog displays extensive help text that's refreshed as the mouse pointer enters each field. This "mouseover" text is the principal source of hardware-specific documentation in TimeLab, since it's always up to date with the latest changes made to each hardware driver. It should be read carefully when using a given instrument or dialog option for the first time.

Miles Design TimePod

The TimePod 5330A uses real-time techniques from the Software-Defined Radio (SDR) field to measure frequency stability, phase noise, and AM noise with specifications substantially beyond those offered by traditional counters and timing analyzers. Depending on the **Available Measurements** options selected in the TimePod acquisition dialog, the 5330A can perform any (or all) measurements supported by TimeLab.

Acquire from counter in Talk-Only mode

This option allows you to acquire frequency, phase/TI, or timestamp readings from almost any GPIB- or RS232-connected counter that supports a "Talk Only" mode, in which ASCII readings are emitted autonomously with no interaction from a host controller. RS232-based timing analyzers such as the picPET are also supported by the Talk-Only option.

TimeLab supports GPIB connectivity through adapters from National Instruments or Prologix. It may also be possible to use later-model Agilent GPIB adapters with NI488.2 compatibility enabled.

Acquire from live ASCII file

Similar to the Talk-Only acquisition option described above, this option allows you to specify the location of a text file containing frequency, phase/TI, or timestamp data that's being written by another process. TimeLab will open the file in read-only mode, process all existing data in the file if requested, and then continue to fetch data until the acquisition terminates. Only one reading per line is processed, but any numeric field within the line can be specified for processing.

HP 53131A/53132A/53181A

HP 53220A/53230A

HP 5335A

HP 5370A/B

HP 5371A/5372A

Philips/Fluke PM6680

Picotest/Array U6200A series

Stanford Research SR620

Wavecrest DTS-2050/2070 series

These instruments are high-performance counters and timing analyzers whose acquisition capabilities are very similar from TimeLab's perspective. Frequency and phase-difference (TI) readings are supported by each. Depending on the instrument's I/O capabilities, available connectivity options may include the following:

- National Instruments NI488.2-compatible GPIB adapters
- Serial (COM) ports connected directly to the instrument
- Serial (COM) ports associated with Prologix GPIB-USB adapters
- IP addresses supported with Prologix GPIB-ETHERNET adapters
- Direct connections to TCP/IP-based instruments at the specified address

HP counters with serial interfaces connected via null-modem cables should use **Acquire→Acquire from counter in Talk-Only mode** instead.

Symmetricom 5115A / 5120A / 5125A (Frequency stability)

Symmetricom 5115A / 5120A / 5125A (Phase noise)

These digital phase noise analyzers provide advanced features and specifications similar to the TimePod 5330A. However, separate options in the **Acquisition** menu must be used to obtain data for xDEV and phase/frequency-difference plots and the Fourier spectrum used for phase noise plots. Phase/frequency data will be acquired continuously for the duration of the measurement, while phase noise acquisition will retrieve a snapshot of the analyzer's current phase noise trace, noise-floor estimate, and spur table.

As with other TimeLab acquisition processes, these acquisitions may be launched concurrently.

Stop/repeat acquisition (Space)

This context-sensitive command associates the most common acquisition "verbs" with a single easy-to-remember keyboard shortcut.

- If deferred-acquisition mode is active and at least one acquisition is pending , the Space bar terminates all active acquisitions (after prompting).
- Otherwise, if the selected plot is associated with an acquisition in progress, the Space bar will terminate that acquisition (after prompting).
- If one or more acquisitions is running but the selected plot is not among them, the Space bar will offer to terminate the acquisition that was most recently started.
- If no acquisitions were running, the Space bar will bring up the most-recently-used acquisition dialog.

Because TimeLab measurement sessions often involve repeated acquisitions with the same instrument, the Space bar shortcut will usually "do what you mean." It will either stop collecting data if an acquisition is in progress, or start a new acquisition if data is not currently being acquired.

Abort and retrigger selected acquisition (Ctrl-a)

Keep and retrigger selected acquisition (Ctrl-k)

Like **Acquire→Stop/repeat acquisition (Space)** , these commands are normally accessed via their respective keyboard shortcuts. Often used when you accidentally disturb the equipment or remember an omitted step in the measurement procedure, Ctrl-a will restart the selected plot's measurement immediately with no further interaction. Ctrl-k behaves similarly, but it will not discard the acquired data before relaunching the acquisition.

These commands operate as follows:

- If no plots are loaded, either command will bring back the most-recently-used acquisition dialog, as if the Space bar shortcut were used.
- Otherwise, if the selected plot is associated with an acquisition in progress, that acquisition will be terminated. (Any other acquisitions in progress will not be affected.)
- If the **Abort (Ctrl-a)** command was issued, the selected plot will be closed and its data discarded. A new acquisition will be started in its place, using the driver associated with the now-deleted plot. The new plot will retain the same position in the legend table.
- If the **Keep (Ctrl-k)** command was issued, the selected plot will *not* be closed. A new acquisition will be launched with its instrument driver. The resulting plot will appear in the next available slot in the legend table.

When applied to an acquisition in progress, *both commands work immediately without prompting for confirmation.* They're assigned to control keys to make them harder to press accidentally.

Enable deferred acquisition (Ctrl-d)

Trigger deferred acquisition(s) (Enter)

Deferred acquisition mode is toggled with the **Enable deferred acquisition (Ctrl-d)** command. In this mode, TimeLab will allow acquisitions to be launched as usual. However, each new measurement's incoming data will be discarded until all pending measurements are *triggered* with the **Trigger deferred acquisition(s) (Enter)** command, or another **Enable deferred acquisition (Ctrl-d)** command is issued to return to immediate-acquisition mode. Data will then be collected normally for the duration of the measurement(s).

Deferred-acquisition mode can be useful when you wish to launch multiple concurrent acquisitions – perhaps using different types of instruments with varying latency or setup time requirements – while synchronizing the beginning of data collection as closely as possible across all measurements. Enable deferred acquisition with Ctrl-d, start the desired measurements, then press Enter to trigger data collection in all acquisitions at once.

Note that TimeLab doesn't specify the trigger latency associated with a group of deferred measurements. Triggered measurements will normally start collecting valid data within one sample period, but this isn't guaranteed.

Configure deferred acquisition

As noted above, when deferred execution is selected, data from subsequent measurements is discarded until the measurement is "triggered" manually with **Trigger deferred acquisition(s) (Enter)** or by cancelling deferred execution with a second **Enable deferred acquisition (Ctrl-d)** command. **Configure deferred acquisition** provides an additional way to trigger a deferred measurement.

With this option, the measurement triggers itself automatically once a specified number of seconds has elapsed after initiation. For example, this feature might be used to ignore the first few minutes of data when measuring oscillators with long warmup times or large startup transients.

When making deferred measurements with the TimePod 5330A, you should consider using scripting techniques instead, as demonstrated in the **OCXO_2hr_stability_test.js** example. Because the 5330A will not tolerate excessive DUT drift while waiting for a deferred measurement to begin, **OCXO_2hr_stability_test.js** delays the entire measurement operation, not just its trigger point.

Help Menu

User guide (F1)

This command launches the system's default Web browser to display http://www.miles.io/timelab/readme.htm . Links on this page will allow you to download the latest version of TimeLab and access the latest edition of the *TimePod 5330A Operation and Service* manual (this file).

When you install TimeLab, a local copy of the TimePod 5330A manual is also installed on your hard drive.

About TimeLab

This command displays an "About" box showing the program version, compilation date, and any supported command-line parameters.

Debug mode

TimeLab contains some debugging/development-specific features that are not documented or supported for general use, and are likely to be removed or deprecated in the future. For example, **Measurement→F)requency spectrum** is intended for debugging, rather than as a supported feature, because the firmware isn't optimized for general spectrum-analysis work. These debugging features will be inaccessible if **Help→Debug mode** is turned off.

It's not recommended that you enable this "debug mode" under most circumstances, as it may disable important safeguards against measurement errors and cause other features to function in unexpected ways.

Check for updates (Ctrl-u)

By default, TimeLab will automatically check the Miles Design web site on a weekly basis and inform you if a newer version is available for download. Updates to TimeLab are always free of charge. You can use the **Help→Check for updates (Ctrl-u)** command to reschedule or disable these notifications.

Appendix: Some examples of residual performance

The gold-standard diagnostic resource for the TimePod 5330A is a simple residual test, conducted after at least a 1-hour warmup, in which you drive the inputs at >= 10 dBm using a clean signal (i.e., without excessive broadband noise) fed through a passive splitter (Mini-Circuits ZFSC-2-1 or similar). The following observations don't reflect specification limits, but they will give you an idea of what to expect.

- At 50 Hz ENBW, residual ADEV at 5 MHz should be in the high E-14s at t=1s. Thermal effects will then push the slope out somewhat, especially in an unstable environment or when warmup time has been insufficient. A look at the **p)hase difference** graph will often show a curved phase slope for the first 30-60 minutes of a measurement made after a cold start.

- Residual PN at 5 MHz should be <= -150 dBc/Hz at 10 Hz after a few hours.

- With a strong, clean input signal at 5 to 10 MHz, the residual PN floor in the 10 kHz - 100 kHz decade will usually end up below -175 dBc/Hz after less than two hours. Expect lower performance near either end of the supported frequency range.

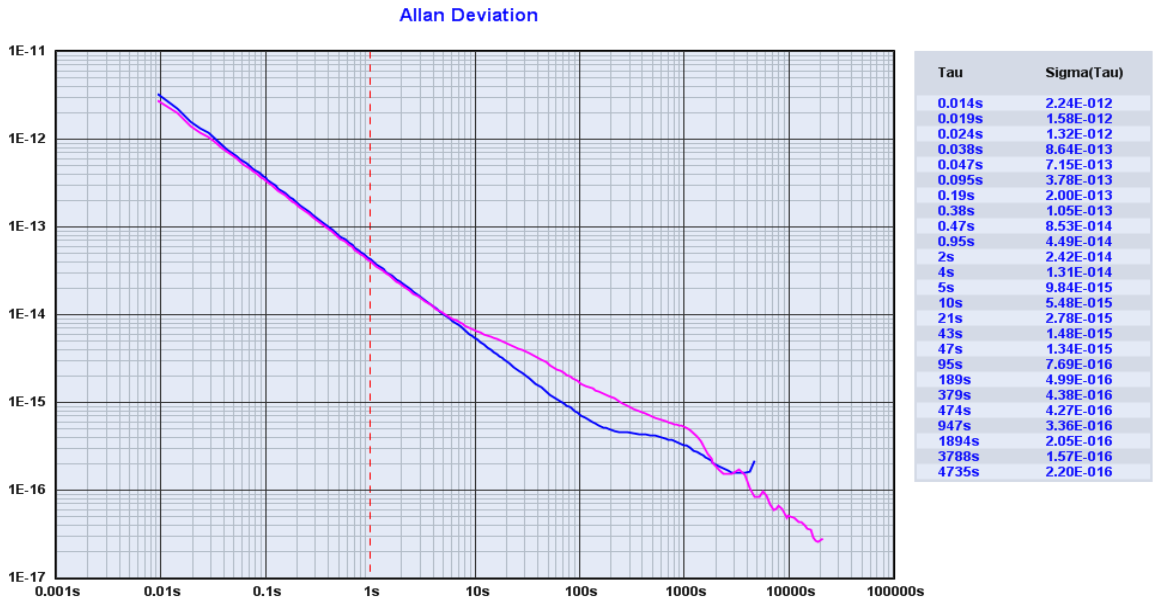
- Phase hits and large outliers in the frequency-difference trace are never normal in a residual test with a clean source signal. They should be investigated if they recur with no obvious explanation.

- Bear in mind that absolute phase-drift measurements require you to explicitly enter the same frequency in both the **Input Freq** and **Reference Freq** fields of the acquisition dialog, unchecking their corresponding **Measure** boxes. Otherwise, the slope of the **p)hase difference** measurement will reflect an arbitrary internal frequency estimate.

As an aid to troubleshooting and performance verification, some typical real-world residual measurements appear below.

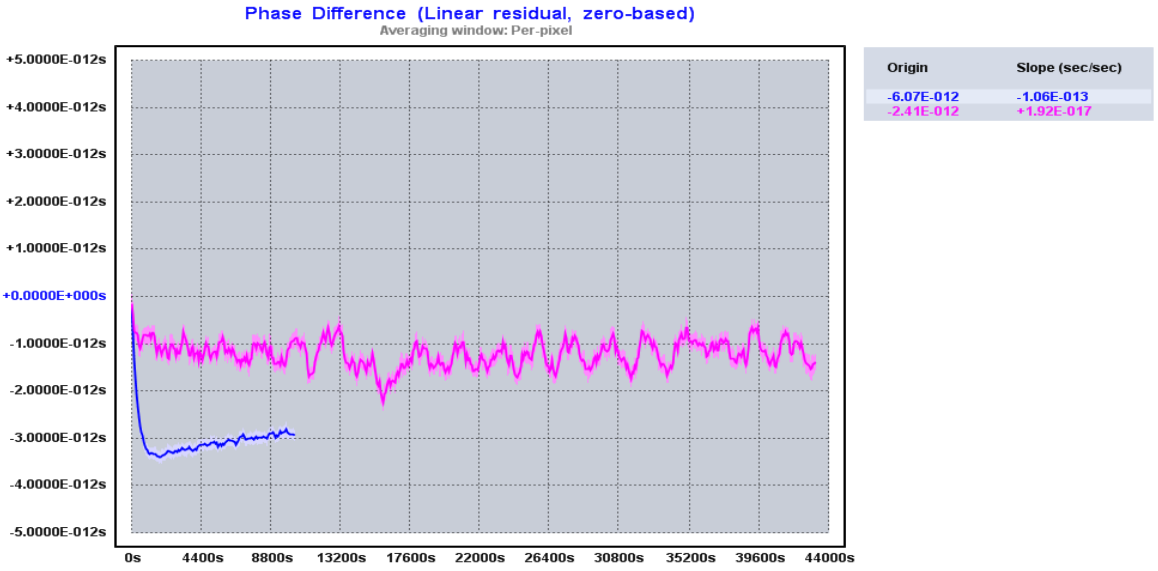
***Blue trace**: 25.68 MHz crystal oscillator, ZFSC-2-1 splitter, about 10 minutes' warmup time. Measurement was allowed to run for three hours.*

***Magenta trace**: 5 MHz crystal oscillator, ZFSC-2-1 splitter, tested under near-ideal conditions with several hours warmup time. Measurement ran overnight (12 hours).*



Tau	Sigma(Tau)
0.014s	2.24E-012
0.019s	1.58E-012
0.024s	1.32E-012
0.038s	8.64E-013
0.047s	7.15E-013
0.095s	3.78E-013
0.19s	2.00E-013
0.38s	1.05E-013
0.47s	8.53E-014
0.95s	4.49E-014
2s	2.42E-014
4s	1.31E-014
5s	9.84E-015
10s	5.48E-015
21s	2.78E-015
43s	1.48E-015
47s	1.34E-015
95s	7.69E-016
189s	4.99E-016
379s	4.38E-016
474s	4.27E-016
947s	3.36E-016
1894s	2.05E-016
3788s	1.57E-016
4735s	2.20E-016

Trace	Input Freq	Input Amplitude	Ref Amplitude	ENBW	ADEV at 1s	Acquired	Instrument
25 MHz residual, 3 hours	25680041.671 Hz	9 dBm	10 dBm	52.7 Hz	4.3E-014	2167272 pts	TimePod 5330A
5 MHz residual, 12 hours	5000000 Hz	12.3 dBm	12.3 dBm	52.7 Hz	4.1E-014	9119124 pts	TimePod 5330A

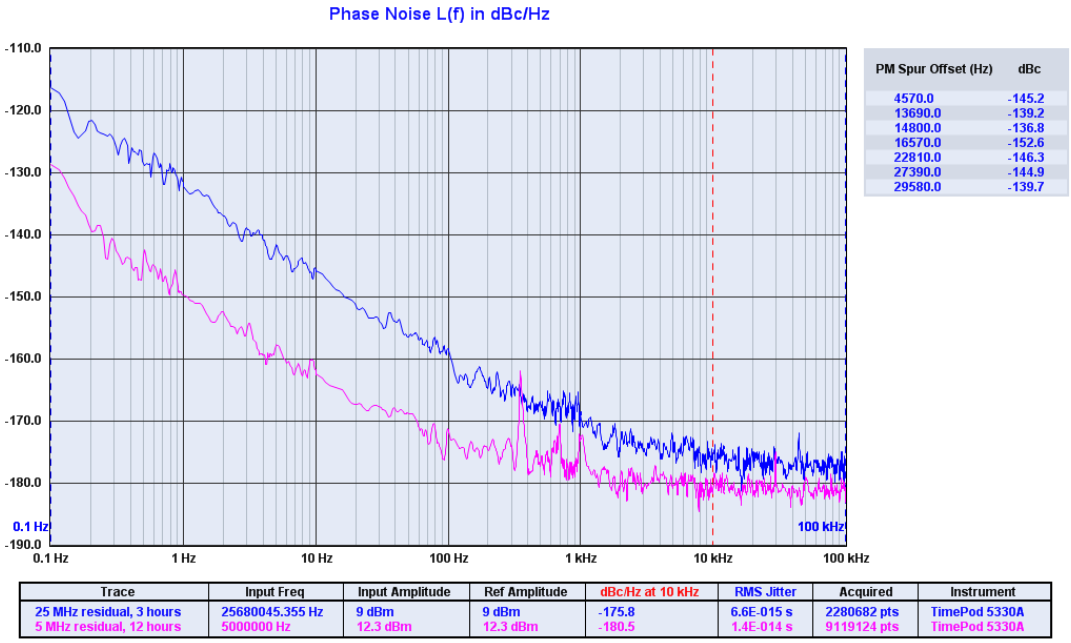


Origin	Slope (sec/sec)
-6.07E-012	-1.06E-013
-2.41E-012	+1.92E-017

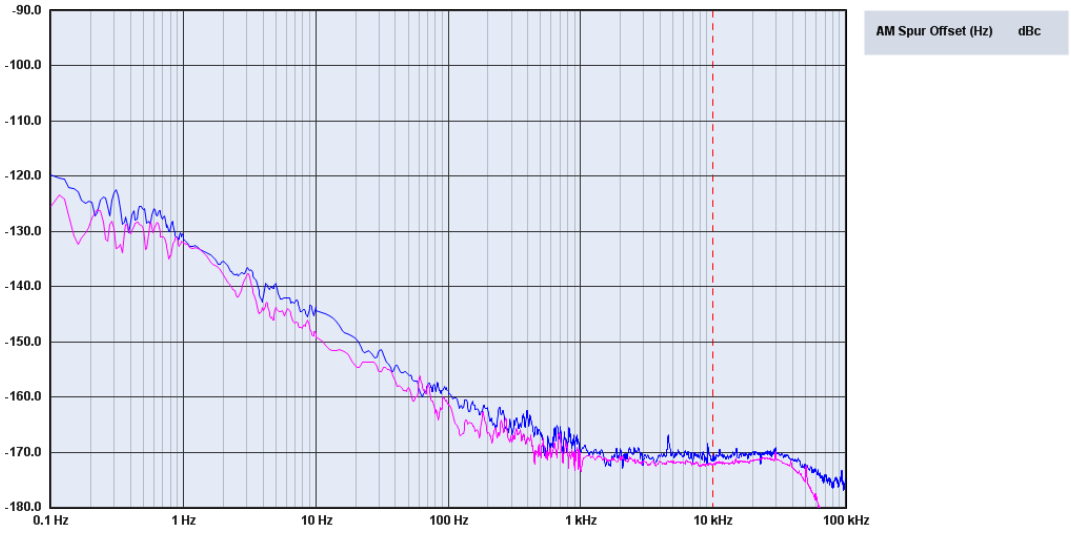
Trace	Input Freq	Input Amplitude	Ref Amplitude	ENBW	Acquired	Instrument
25 MHz residual, 3 hours	25680041.671 Hz	9 dBm	10 dBm	52.7 Hz	2167272 pts	TimePod 5330A
5 MHz residual, 12 hours	5000000 Hz	12.3 dBm	12.3 dBm	52.7 Hz	9119124 pts	TimePod 5330A

Despite visible HVAC cycling, overall phase stability is within +/- 1 ps over the entire 12-hour 5 MHz run.

The 25 MHz residual trace, on the other hand, was taken with inadequate warmup time. It begins with about 30 minutes of noticeable phase drift (about three picoseconds) until the TimePod's internal components reach thermal equilibrium. Slow drift in the opposite direction is then observed.



Trace	Input Freq	Input Amplitude	Ref Amplitude	dBc/Hz at 10 kHz	RMS Jitter	Acquired	Instrument
25 MHz residual, 3 hours	25680045.355 Hz	9 dBm	9 dBm	-175.8	6.6E-015 s	2280682 pts	TimePod 5330A
5 MHz residual, 12 hours	5000000 Hz	12.3 dBm	12.3 dBm	-180.5	1.4E-014 s	9119124 pts	TimePod 5330A



Trace	Input Freq	Input Amplitude	Ref Amplitude	dBc/Hz at 10 kHz	Acquired	Instrument
25 MHz residual, 3 hours	25680045.355 Hz	9 dBm	9 dBm	-170.8	2280682 pts	TimePod 5330A
5 MHz residual, 12 hours	5000000 Hz	12.3 dBm	12.3 dBm	-172.2	9119124 pts	TimePod 5330A

As expected, residual phase noise performance is improved when a strong 5 MHz signal is measured for 12 hours, compared to a weaker 25 MHz signal measured for 3 hours. The difference in AM noise isn't as pronounced in this case.

Appendix: JavaScript API Function Reference

TimeLab includes an embedded JavaScript engine[14] that can run automated test scripts using the TimePod 5330A. (Script-based access to other instruments such as time-interval counters is not currently supported.) The API functions listed below are useful in a variety of applications ranging from the standard TimePod performance verification procedures to user-developed test scripts.

Usage examples for nearly all of these API functions can be found in the scripts included with TimeLab, such as the standard validation/diagnostic scripts **5330A_performance_test.js** and **5330A_ADC_test.js** and the simple example program **OCXO_2hr_stability_test.js**. For more information on creating, editing, and running test scripts, refer to the **Scripts** menu description on page 99.

AcqChannel(Number channel)

Returns the position in the legend table beneath the graph, from 0 to 8, of the plot associated with the script instance's specified acquisition channel. Since plots being acquired under script control may be moved up and down in the legend table by the user like any other loaded plots, **AcqChannel()** is needed to obtain the value needed when calling **ScriptBindToPlot()** in multichannel acquisitions.

The **channel** parameter should range from 0 to one less than the value returned by **AcqNumChannels()**. Scripts that perform only standard single-channel acquisitions will not need to use this function.

AcqCheckOptions(String option_name [{, String option_name}])

This function is reserved for compatibility with Symmetricom 3120A test scripts. It always returns TRUE in scripts associated with TimePod 5330A measurements.

AcqClearParams()

Each script instance maintains a dictionary of acquisition parameters that can be used to override any or all of the acquisition dialog parameters when **AcqStartAcquisition()** is called. **AcqClearParams()** clears this dictionary. It is normally called before issuing the **AcqSetParam()** call(s) needed to configure the measurement.

Acquisition parameters that the script does not explicitly set with **AcqSetParam()** will assume their default values when the measurement begins.

See **AcqParams()** and **AcqSetParam()** for more information.

[14] http://code.google.com/p/v8/

AcqGetAMSpurTable(Array offset_array, Array amplitude_array)

Populates **offset_array** and **amplitude_array** with a list of all detected AM spurs in the current measurement, returning the total number of AM spurs.

AcqGetPMSpurTable(Array offset_array, Array amplitude_array)

Populates **offset_array** and **amplitude_array** with a list of all detected PM spurs in the current measurement, returning the total number of PM spurs.

AcqGetXDEVBins(String selection, Array tau_array, Array sigma_array)

AcqGetXDEVChart(Array tau_array, Array sigma_array)

Populates **tau_array** and **sigma_array** with a list of sigma(tau) values from all valid bins in the specified statistical measurement, or from the most recently rendered sigma(tau) chart, depending on which function is called. The selection parameter passed to **AcqGetXDEVBins()** should be a string beginning with 'a', 'h', 'm', or 't', corresponding to Allan deviation, Hadamard deviation, modified Allan deviation, or time deviation, respectively.

AcqGetXDEVBins() differs from **AcqGetXDEVChart()** in that **AcqGetXDEVChart()** returns results only for the 1, 3, 10 tau multiples beginning at t=1s that were displayed in the most-recently-visible numeric table for the measurement. Before calling **AcqGetXDEVChart()**, you should use **MeasurementSelectView()** or **MeasurementDeferSelectView()** to display the statistical measurement for which the chart contents should be returned. Additionally, **TraceShowNumTab(true)** should be called to ensure that the numeric table is visible, and the script must allow time for at least one frame to be rendered. **AcqGetXDEVBins()** does not have any of these constraints.

AcqNumAcquiredPoints()

Returns the number of acquired points in the phase data record.

AcqNumChannels()

Returns the number of acquisition channels in the measurement associated with the script instance.

In scripts that perform only standard single-channel acquisitions, this function always returns 1.

AcqParam(String key)

Each script instance maintains a dictionary of acquisition parameters that can be used to override any or all of the default acquisition dialog parameters when **AcqStartAcquisition()** is called. **AcqParam()** returns the value for the specified acquisition parameter.

If an existing plot is associated with the script instance, **AcqParam()** will return the specific value associated with the plot. Otherwise, if the script has not yet started a measurement, **AcqParam()** will return the value from the dictionary of initial parameters that will be passed to **AcqStartAcquisition()**. for the plot. In either case, an attempt to look up an unrecognized parameter value will return an empty string.

See **AcqSetParam()** and **AcqClearParams()** for more information.

AcqReadAMNoiseTrace(Number offset_Hz)

AcqReadPMNoiseTrace(Number offset_Hz)

These functions return the current AM noise or phase noise level in dBc/Hz at the specified offset from the carrier. A value greater than 0 indicates that data is unavailable, either because the AM noise or phase noise trace has not yet been displayed or because noise data from the FFT segment containing the specified offset frequency is not yet available.

Values returned by these functions are based on the most recent AM noise or phase noise trace rendered. Scripts should use **MeasurementSelectView()** or **MeasurementDeferSelectView()** to select the appropriate measurement view prior to attempting to read noise trace values. Typically this is done from within an **EventTimer()** event handler, in order to ensure that valid data is available during a subsequent timer event.

AcqSetParam(String key, String value)

Each script instance maintains a dictionary of acquisition parameters that can be used to override any or all of the acquisition dialog parameters when **AcqStartAcquisition()** is called. **AcqSetParam()** updates the stored value for the specified acquisition parameter.

The best way to obtain a list of measurement parameter names and values for a given instrument is to use the **Save Measurement Script** button in the **Utility** tab of the acquisition dialog to create a script that will contain explicit **AcqSetParam()** statements for each supported parameter. Almost all parameters will have obvious counterparts in the instrument's acquisition dialog; you can also refer to the **Legend** menu section on page 93 for individual descriptions.

See **AcqParam()** and **AcqClearParams()** for more information.

AcqStartAcquisition(String menu_entry, Boolean skip_dialog)

Starts a new measurement based on the script instance's current set of acquisition parameters.

menu_entry should correspond *exactly* to the name of the instrument as it appears on the **Acquire** menu, including any ellipsis (...) that follows it. Currently the only officially-supported instrument name for scripted measurements is the *Miles Design TimePod ...* entry, so this string should be supplied as the **menu_entry** value.

With a script-initiated measurement, the acquisition parameters are recalled from the same driver-specific .INI file that provides the dialog defaults when the acquisition is initiated manually from the **Acquire** menu. These default parameters are then overwritten with any entries that appear in the script instance's acquisition-parameter dictionary, as described in **AcqParam**() and **AcqSetParam**(). (Unlike a manual measurement, script-based measurements do not write their parameters back to the driver's .INI file. As a result, the dialog defaults that will appear in the next user-initiated acquisition will not reflect the parameters used for the last scripted acquisition.)

Next, if **skip_dialog** is false, TimeLab will present the acquisition dialog to the user, allowing any parameters to be changed before the measurement begins. In most cases, **skip_dialog** should be set to true to allow the script to run without further user intervention. This will be the case for measurement scripts created with the **Save Measurement Script** button in the acquisition dialog's **Utility** tab.

AcqStartAcquisition() will return true if the acquisition is initiated successfully, or false if an error occurred. Reasons for failure might include the lack of an available slot in the legend table for the plot(s) created by the acquisition, the use of an unrecognized **menu_entry** name, the specification of an invalid acquisition parameter value with **AcqSetParam(),** or (if **skip_dialog** is false) the user's selection of the **Cancel** button.

Once the measurement begins collecting data, the script's **EventAcqTriggered()** event handler (if any) will be called. Because **AcqStartAcquisition()** returns immediately after launching the acquisition thread, measurement-specific parameters such as **Input Freq** will not be available to the script until **EventAcqTriggered()** is called.

AcqStopAcquisition()

Immediately stops any acquisition associated with the running script instance and calls the script's **EventAcqDone()** handler (if any). Unless a fatal error occurs, the plot(s) will not be closed and the script will continue to run until **ScriptEnd()** is called.

DisplayOverlayMode([Boolean overlay_mode])

Equivalent to the **Display→Browse plots one at a time** (b) or **Display→Overlay all loaded plots** (o) command, depending on the **overlay_mode** parameter.

overlay_mode is optional; if it is not provided by the script, the function simply returns true in **Overlay** mode or false in **Browse** mode. If a new **overlay_mode** value is specified, the function returns the previous display mode.

DisplayShowNumTab([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Display→Numeric table** (Ctrl-n). Refer to this command for more information.

To return the current flag value, call **DisplayShowNumTab()** with no argument.

EventRunScript()

This is a mandatory user-supplied function that must appear in every TimeLab script. It is called when the user issues **Scripts→Run script** or a similar command, and serves as the entry point for script execution.

Typically, **EventRunScript()** does little more than set up the appropriate acquisition parameters and call **AcqStartAcquisition()**, returning immediately afterwards. Subsequently, the script's **EventAcqTriggered()** handler is called by TimeLab when the acquisition begins collecting data. **EventAcqTriggered(),** in turn, may use **TimeSetTimer()** to arrange for periodic callback service to **EventTimer()**. Scripts that need to perform extensive measurement supervision and control operations normally do so within their **EventTimer()** handlers, while final result evaluation and report generation is done within **EventAcqDone()**.

For a detailed look at the operation and overall life cycle of TimeLab scripts, refer to the **5330A_performance_test.js**, **5330A_ADC_test.js**, and **OCXO_2hr_stability_test.js** example scripts provided with TimeLab, as well as the other API function descriptions in this section.

EventAcqTriggered()

This is an optional user-supplied function. EventAcqTriggered() is called when a measurement initiated by AcqStartAcquisition() begins collecting data. Typically this happens a few seconds after AcqStartAcquisition() is called by the user-provided EventRunScript() function, unless the deferred acquisition options on the **Acquire** menu have been used to postpone data collection.

Any errors that occur between EventRunScript() and EventAcqTriggered() automatically terminate script execution.

See EventRunScript() for further information.

EventAcqDone([success])

This is an optional user-supplied function. EventAcqDone() is called when an acquisition ends, either because the requested measurement duration expired, because an error occurred, or because it was stopped manually by the user. The optional success parameter is a Boolean value that indicates whether the hardware driver reported any errors during the measurement.

See EventRunScript() for further information.

EventTimer()

This is an optional user-supplied function. EventTimer() is called periodically at a rate determined by the parameter passed to TimeSetTimer().

Because TimeLab's scripting system relies on cooperative multitasking, most non-trivial scripts should carry out the majority of their processing work in their EventTimer() handler while their measurement(s) are in progress. For detailed examples of timer usage, refer to **5330A_performance_test.js** and **5330A_ADC_test.js**. Also see EventRunScript() for further information.

FileCloseAllChannels()

Closes all plot(s) acquired in the most recent measurement executed by the script, stopping any acquisition(s) that may still be in progress. For conventional single-channel measurements, FileCloseAllChannels() is equivalent to calling FileClosePlot() with no argument.

As an example, **5330A_performance_test.js** uses FileCloseAllChannels() in its EventAcqDone() handler to discard the two plots generated during the preliminary warmup acquisition that the script performs before running the actual test. (Because the warmup acquisition has already terminated at this point, EventAcqDone() is not reissued.) Simply calling FileClosePlot() here would have closed only the first of the two plots.

FileCloseAllPlots()

Equivalent to calling **FileClosePlot()** on all loaded plots. For example, **5330A_ADC_test.js** uses **FileCloseAllPlots()** to ensure that enough slots are available in the legend table to hold the four plots that it acquires for various ADC combinations.

FileClosePlot([Number selection])

If the optional **selection** parameter is specified, **FileClosePlot()** closes the plot at the specified position in the legend table, from 0 to 8. Since plots being acquired under script control may be moved up and down in the legend table by the user like any other loaded plots, **AcqChannel()** is needed to obtain the value passed to this function if the intent is to close any of the plot(s) that are associated with a measurement launched by the script.

Alternatively, **selection** may be omitted. In this case, the function closes the plot associated with the first channel of the most recent measurement executed by the script.

In both cases, calling **FileClosePlot()** on a plot associated with a given measurement will stop any acquisition(s) in progress that were initiated by that measurement. The script's **EventAcqDone()** handler, if any, will be called for any acquisitions that were terminated by **FileClosePlot()**.

FileExecute(String path_to_file [, String params])

FileExecute() attempts to execute or otherwise open the specified file using the "open" verb with the Win32 ShellExecute() API function. If ShellExecute() returns successfully, **FileExecute()** returns true; otherwise, if an error occurred, it returns false.

path_to_file should be a fully-qualified pathname. Optionally, a string consisting of one or more parameters may be passed to ShellExecute() via the **params** argument.

As an example, **FileExecute()** is used by **5330A_performance_test.js** to view the generated HTML report page on the system's Web browser.

FileNumLoadedPlots()

Returns the number of currently-loaded plots, from 0 to 9.

FileNumUnsavedPlots()

Returns the number of currently-loaded plots, from 0 to 9, that have not been saved since being acquired or modified.

FileSave(String path, [String subdir,] String filename)

Based on the (case-insensitive) suffix of **filename**, **FileSave()** saves either a screen image in .TGA, .GIF, .BMP, .PCX, or .PNG format, or a .TIM file representing the measurement currently associated with the script.

By default, if **path** is empty, the user's Documents folder is treated as the destination path. This path must already exist. The **subdir** parameter is optional; it represents a subdirectory relative to **path** in which the destination file will be saved. Unlike **path**, the **subdir** folder will be created automatically if it does not already exist.

FileSave() returns a string containing the saved file's fully-qualified pathname. Any errors that occur will not be reported to the script. They will result in a JavaScript exception which will terminate the script with an appropriate error message.

To save .TIM data from multiple plots in a multichannel measurement, you must use **ScriptBindToPlot()** to associate the script with channels other than the first one. See **5330A_performance_test.js** for an example.

FileSaveText(String path, [String subdir,] String filename, String contents)

FileSaveText() saves the ASCII **contents** string to **filename**. As with **FileSave()** above, the user's Documents folder is treated as the destination folder if **path** is empty. Any explicitly-specified **path** must already exist. The **subdir** parameter is optional; it represents a subdirectory relative to **path** in which the destination file will be saved. Unlike **path**, the **subdir** folder will be created automatically if it does not already exist.

FileSaveText() returns a string containing the saved file's fully-qualified pathname. Any errors that occur will not be reported to the script. They will result in a JavaScript exception which will terminate the script with an appropriate error message.

As an example, **5330A_performance_test.js uses** **FileSaveText()** to save its HTML report.

LegendSelect([String field_name] [,] [Boolean status])

Controls and/or reports the visibility of a given parameter (**field_name**) in the legend table beneath the graph.

This function's arguments may consist of a single Boolean value, in which case it will select or deselect *all* legend fields in a manner similar to the **Legend→All** or **Legend→None** menu commands; a single String value, which will simply return the visibility state of the specified **field_name** without changing it; or both String and Boolean parameters, which will show (**status**==true) or hide (**status**==false) the specified **field_name** in the table and return its previous state.

MaskResultMargin()

Returns a Number value representing the margin by which the measurement associated with the script is passing (if positive) or failing (if negative) the mask test selected by **MaskSelect()**. The returned value is based on the most recently rendered frame, so it is dependent on both the current measurement view *and* the currently selected mask. Refer to the **Masks** menu description on page 97 for more information about TimeLab's mask-test functionality.

Important: Keep in mind that JavaScript execution does not disable or inhibit the rest of the TimeLab user interface. While test masks are easy to create and use in TimeLab, script-based mask evaluation is complicated by the need to establish the correct measurement view, allow enough time for the display to update, and wait for valid results to become available for the X-axis range covered by the mask, all while allowing for the user's ability to switch measurement views, move plots up and down in the legend table, or alter other settings manually at any time.

Typically the best way to coordinate the necessary script actions while allowing for manual user intervention is to use an **EventTimer()** handler to check the mask results and the current measurement view selection at the same time, then make any changes (such as selecting the next measurement view for evaluation) at the very end of the timer handler. Script authors are *strongly* encouraged to use the **EventTimer()** handler in **5330A_performance_test.js** as a model for their own mask-test procedures.

MaskResultValid()

Returns true if valid data is available from **MaskResultMargin()**, or false if the mask test was not ready for evaluation for any reason. For example, ADEV mask results will not be available if the measurement has not yet run long enough to return valid data at the tau corresponding to the left endpoint of the mask limit line, or if the user has turned visibility off for the plot associated with the script. Always check **MaskResultValid()** prior to calling **MaskResultMargin()**.

See **MaskResultMargin()** for other notes on scripted mask tests.

MaskSelect([String mask_name])

This function selects a mask for use with **MaskResultMargin()** and **MaskResultValid()**, returning the previously-selected mask if any. If **mask_name** is omitted, the function returns the currently-selected mask name. In both cases, the function will return an empty string if no mask was selected.

Mask names should be specified *exactly* as they appear in the **masks.txt** file accessed via **Masks→Edit mask definitions** (which also determines the names under which they are added to the **Masks** menu). See **MaskResultMargin()** for other notes on scripted mask tests.

MeasurementSelectView(String hotkey)

MeasurementDeferSelectView(String hotkey)

These functions accept a **hotkey** parameter that consists of the (case-sensitive) shortcut key for the desired measurement view as it appears in the **Measurement** menu. For example, **MeasurementSelectView("P")** will emulate the 'P' keyboard shortcut (**Measurement→Phase noise (P)**).

If your script is performing mask tests with multiple measurement types, it's a good idea to call **MeasurementDeferSelectView()** as the last action taken before returning from **EventTimer()**. Upon the next invocation of **EventTimer()**, your script should call **MeasurementView()** and evaluate the masks or other test parameters for that view, regardless of any changes to the measurement view that may have taken place between timer ticks. Script authors are encouraged to use the **EventTimer()** handler in **5330A_performance_test.js** as a model for test procedures that use multiple measurement types.

The difference between **MeasurementSelectView()** and **MeasurementDeferSelectView()** is that the latter will not take effect until after the event handler returns. This distinction is important when multiple instances of the same script may be executed at once. Essentially, deferring the view selection until after all script instances with the same **EventTimer()** interval have been serviced leaves a single script instance "in charge" of the view, reducing opportunities for confusion.

MeasurementView()

Returns the current measurement view as a String containing the hotkey for the view as it appears in the **Measurement** menu. For example, when viewing a phase noise plot, **MeasurementView()** will return "P", the shortcut key for **Measurement→Phase noise**.

See **MeasurementSelectView()** for additional notes on measurement view selection.

Print(…)

Displays one or more user-specified values in the script console window associated with the currently-bound plot. Multiple arguments may be separated by commas; a line feed will be inserted after each argument. (To avoid this, you can use string concatenation rather than multiple Print() arguments.)

Numerous usage examples for the Print() function appear in **5330A_performance_test.js** and **5330A_ADC_test.js**.

ScriptBindToPlot(Number selection)

Associates the script instance with the plot at the specified position in the legend table, from 0 to 8.

Most script authors will not need to use ScriptBindToPlot(). It is required only in scripts that support multichannel plots, such as **5330A_performance_test.js** and **5330A_ADC_test.js**. A script can be associated ("bound") to only one plot at a time, and most operations performed by a script implicitly involve the currently bound plot. When a measurement initiated by a script specifies multiple stability channels, as in the TimePod 5330A validation scripts mentioned, it's necessary to use ScriptBindToPlot() to access the plots that represent all channels of the measurement.

By default, scripts are bound to channel 0 of a multichannel acquisition. It's a good idea to restore the default channel-0 binding with ScriptBindToPlot(AcqChannel(0)) before returning from any function or event handler that uses ScriptBindToPlot().

Since plots in multichannel measurements may be moved up and down in the legend table by the user, ScriptBindToPlot() is normally used together with the AcqChannel() function. See the two performance test scripts mentioned above for further comments and usage examples.

ScriptBoundToPlot()

Returns the position in the legend table beneath the graph, from 0 to 8, of the plot to which the script is currently bound.

Most script authors will not need to use ScriptBoundToPlot(). It is required only in scripts that support multichannel plots, such as **5330A_performance_test.js** and **5330A_ADC_test.js**.

ScriptEnd()

Marks the script for cleanup by the TimeLab runtime JavaScript engine. Scripts which are associated with loaded plots must be terminated with a call to **ScriptEnd()** when finished. Otherwise, they will continue to run (and consume resources) until one of the following conditions is true:

- A script exception or other fatal error occurs
- The **Scripts→Stop all running scripts (**F12**)** command is issued by the user
- The user exits from TimeLab

ScriptFilename(Boolean full_path)

If **full_path** is true, ScriptFilename() returns a String containing the fully-qualified path to the script's JavaScript (.js) source file. Otherwise, it returns only the filename itself.

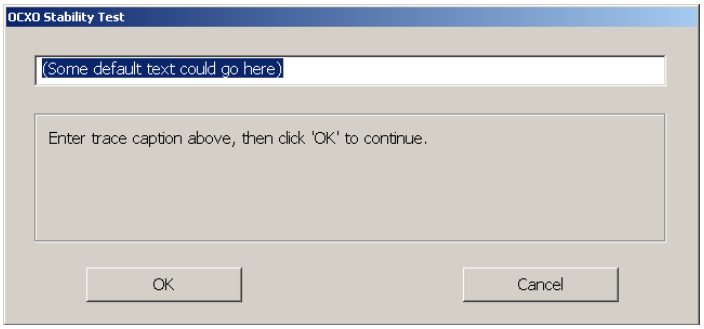
ScriptGetString(String heading, String message, Object user_text, String b1_text [, String b2_text [, String b3_txt]])

Presents a modal dialog box to the user that accepts freeform text entry. An example appears below.

```
var trace_caption = { value:"(Some default text could go here)" };

if (ScriptGetString(   "OCXO Stability Test",
                       "Enter trace caption above, then click 'OK' to continue.",
                        trace_caption,
                       "OK",
                       "Cancel") != 0)
       {
       ScriptEnd();
       return;
       }
```

When executed, the result is a dialog box of the form



Text for one, two, or three buttons can be specified as the **b1_text**, **b2_text**, and/or **b3_text** argument(s) to **ScriptGetString()**. The returned value will equal the numeric index of the button selected by the user, beginning with 0 for the **b1_text** button.

All text fields accept up to 1024 bytes. Note that the **user_text** argument is an Object containing a String property called **value**, not a String in itself. The **value** property is used both as the default text used to initialize the edit control and for reception of the returned text. (This is necessary because Strings are always passed by copy in JavaScript, while Objects are passed by reference and may be modified by the called function.)

See **OCXO_2hr_stability_test.js** for a more complete usage example.

ScriptHostFilename()

Returns a String containing the TimeLab executable filename and version information.

ScriptLastStatusMessage()

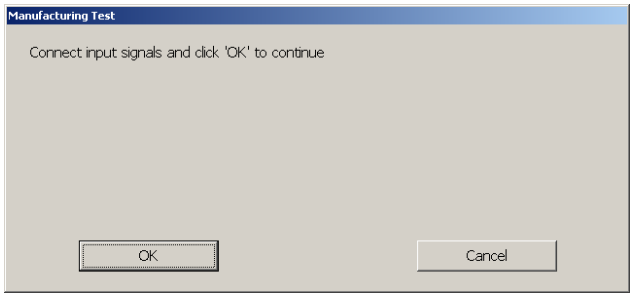
Returns a String containing the most recent status message displayed by TimeLab. This function may be useful for logging and report generation.

ScriptMessageBox(String heading, String message, String b1_text [, String b2_text [, String b3_txt]])

Presents a modal dialog box to the user that accepts button presses. An example appears below.

```
if (ScriptMessageBox(  "Manufacturing Test",
                       "Connect input signals and click 'OK' to continue.",
                       "OK",
                       "Cancel") != 0)
       {
       ScriptEnd();
       return;
       }
```

When executed, the result is a dialog box of the form



Text for one, two, or three buttons can be specified as the **b1_text**, **b2_text**, and/or **b3_text** argument(s) to **ScriptMessageBox()**. The returned value will equal the numeric index of the button selected by the user, beginning with 0 for the **b1_text** button.

ScriptMinsToHMS(Number minutes)

Returns a String representing the **minutes** argument as hours, minutes, and seconds.

Example: The following statement

```
Print(ScriptMinsToHMS(10.5));
```

will display:

```
10m 30s
```

ScriptNumVal(String format_string, Number value)

Returns a String consisting of the specified **format_string** with an embedded numeric **value.**

In operation, **ScriptNumVal()** passes its **format_string** and floating-point **value** arguments to the standard C sprintf() function, returning the resulting formatted string.

Example: The following statement

```
Print(ScriptNumVal("Pi = %.2lf", 3.1415926535));
```

will display:

```
Pi = 3.14
```

Additional usage examples appear in both **5330A_performance_test.js** and **5330A_ADC_test.js**. Refer to the sprintf() documentation in the C runtime library reference for a detailed discussion of the format and precision specifiers supported for floating-point values.

ScriptPlaySound(String filename)

Plays the specified sound file. Sound files for use with this function are typically stored in the .wav format.

If **filename** contains a backslash character, TimeLab assumes it consists of a fully-qualified path name and plays the sound from the filename exactly as specified. Otherwise, filenames without backslash characters are played from the TimeLab installation directory.

ScriptSetTTYHeader(Number linenum [, String text])

Sets and/or retrieves the text displayed at the top of the script TTY window that is associated with the plot(s) bound to the script instance.

The **linenum** argument specifies which of the first two TTY lines will be retrieved or updated; it must equal 0 or 1. The **text** argument is optional. If present, it will replace the existing text for the specified TTY line. Regardless, the function will return the contents of the specified line at the time it was called.

Note that the script TTY is associated with the most recent plot that has been acquired by the script or otherwise bound to it. (All channels in a multichannel acquisition share the same TTY.) The **Print** function will write its output to the TimeLab status line if no TTY is associated with the script, but **ScriptTTYHeader()** will fail with an exception. In most cases, you should not attempt to write to the TTY before **AcqStartAcquisition()** has been called.

ScriptShowTTY([Boolean show])

Shows or hides the script TTY window that is associated with the plot(s) bound to the script instance, returning the previous visibility state. If the **show** argument is omitted, the function will return the current visibility state without changing it.

The script TTY window becomes visible by default upon successful execution of **AcqStartAcquisition()**. However, because the TTY can obscure a large portion of the TimeLab graph display area, simple scripts such as **OCXO_2hr_stability_test.js** that report little or no status information should hide the TTY window. The user will still have the option to select **Scripts→Toggle script console for selected plot (**F11**)** to show the TTY manually.

Unlike the other TTY functions, **ScriptShowTTY()** may be called at any time. The visibility state that it establishes will be inherited by any subsequent plots that the script acquires.

ScriptShowTTY([Boolean show])

Shows or hides the script TTY window that is associated with the plot(s) bound to the script instance, returning

ScriptShowTTY([Boolean show])

Shows or hides the script TTY window that is associated with the plot(s) bound to the script instance, returning

TimeHoursSinceTrigger()

TimeMinsSinceTrigger()

TimeSecsSinceTrigger()

These three functions return the elapsed time since the measurement currently associated with the script was triggered, if the measurement is still in progress. Otherwise, if the measurement is no longer in progress, the overall duration of the acquisition is returned.

The values returned are floating-point Numbers that equivalently represent the elapsed time in terms of hours, minutes or seconds.

TimeSetTimer(Number msec)

This function arranges for TimeLab to call the script's **EventTimer()** handler at regular intervals of **msec** milliseconds.

Because TimeLab's scripting system relies on cooperative multitasking, most non-trivial scripts should carry out the majority of their processing work in their **EventTimer()** handler while their measurement(s) are in progress. For detailed examples of timer usage, refer to **5330A_performance_test.js** and **5330A_ADC_test.js**.

Prior to termination, a script that uses timer service should call **TimeSetTimer(0)** to discontinue further callbacks.

TraceAvgWindow([Number seconds])

Sets and/or retrieves the duration in **seconds**, from 0.01 to 10000.0, of the averaging window used for phase- and frequency-difference plots. An argument of 0 **seconds** disables averaging entirely.

The function returns the previous window duration. It may be called with no argument to return the current value without altering it.

See the **Trace→Averaging window for phase/frequency traces (g)** command for important information on trace averaging.

TraceMarkSpurs([Boolean status])

Sets and/or retrieves the **status** of the spur-visibility flag controlled by **Trace→Mark spurs in noise traces (Ctrl-m)**. Refer to this command for more information.

To return the current spur-visibility flag value, call **TraceMarkSpurs()** with no argument.

TraceOverlayAMPM([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show AM noise in PN view (F8)**. Refer to this command for more information.

To return the current flag value, call **TraceOverlayAMPM()** with no argument.

TracePhaseFreqAutoY([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Phase/frequency Y axis unlocked in zoom mode (y)**. Refer to this command for more information.

To return the current flag value, call **TracePhaseFreqAutoY()** with no argument.

TracePhaseFreqResid([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show linear phase/frequency residual (r)**. Refer to this command for more information.

To return the current flag value, call **TracePhaseFreqResid()** with no argument.

TracePhaseFreqTrend([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show linear phase/frequency trend (Ctrl-t)**. Refer to this command for more information.

To return the current flag value, call **TracePhaseFreqTrend()** with no argument.

TracePhaseFreqZero([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Phase/frequency traces begin at zero (z)**. Refer to this command for more information.

To return the current flag value, call **TracePhaseFreqZero()** with no argument.

TraceShowCorrGain([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show correlation gain for selected noise trace (Ctrl-g)**. Refer to this command for more information.

To return the current flag value, call **TraceShowCorrGain()** with no argument.

TraceShowImag([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show imaginary part of cross spectrum (Ctrl-F3)**. Refer to this command for more information.

To return the current flag value, call **TraceShowImag()** with no argument.

TraceShowNoiseFloor([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show estimated instrument noise (F2)**. Refer to this command for more information.

To return the current flag value, call **TraceShowNoiseFloor()** with no argument.

TraceShowRaw([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show raw PN channel trace(s) (Ctrl-r)**. Refer to this command for more information.

To return the current flag value, call **TraceShowRaw()** with no argument.

TraceShowSlopes([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show FFT segment filter slope(s) (Ctrl-i)**. Refer to this command for more information.

To return the current flag value, call **TraceShowSlopes()** with no argument.

TraceSmoothNoise([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Smooth noise traces (Ctrl-w)**. Refer to this command for more information.

To return the current flag value, call **TraceSmoothNoise()** with no argument.

TraceThickness([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Toggle trace thickness for current measurement (T)**. Refer to this command for more information.

To return the current flag value, call **TraceThickness()** with no argument.

TraceTickMarks([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Tick marks (k)**. Refer to this command for more information.

To return the current flag value, call **TraceTickMarks()** with no argument.

TraceXDEVBars([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Show xDEV error bars (Ctrl-e)**. Refer to this command for more information.

To return the current flag value, call **TraceXDEVBars()** with no argument.

TraceXDEVClipBW([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Clip xDEV traces by noise bandwidth (Ctrl-b)**. Refer to this command for more information.

To return the current flag value, call **TraceXDEVClipBW()** with no argument.

TraceXDEVClipConfidence([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Clip xDEV traces by confidence (Ctrl-v)**. Refer to this command for more information.

To return the current flag value, call **TraceXDEVClipConfidence()** with no argument.

TraceXDEVSpline([Boolean status])

Sets and/or retrieves the **status** of the flag controlled by **Trace→Draw xDEV traces with spline interpolation (i)**. Refer to this command for more information.

To return the current flag value, call **TraceXDEVSpline()** with no argument.

Appendix: The STREAM.EXE Phase/Frequency Data Server

TimeLab includes a Windows console application, **STREAM.EXE**, which can be used to provide remote access to measurement data acquired by a TimePod 5330A. Continuous streams of phase-difference or frequency measurements acquired by **STREAM.EXE** can be written to a shared file or streamed continuously via TCP/IP over a local- or wide-area network.

Launching STREAM.EXE

STREAM.EXE uses relatively little CPU time or RAM compared to the full-fledged TimeLab application. As a console application, it displays no graphics at all. Consequently it can run on almost any Windows PC, including energy-efficient servers or netbooks.

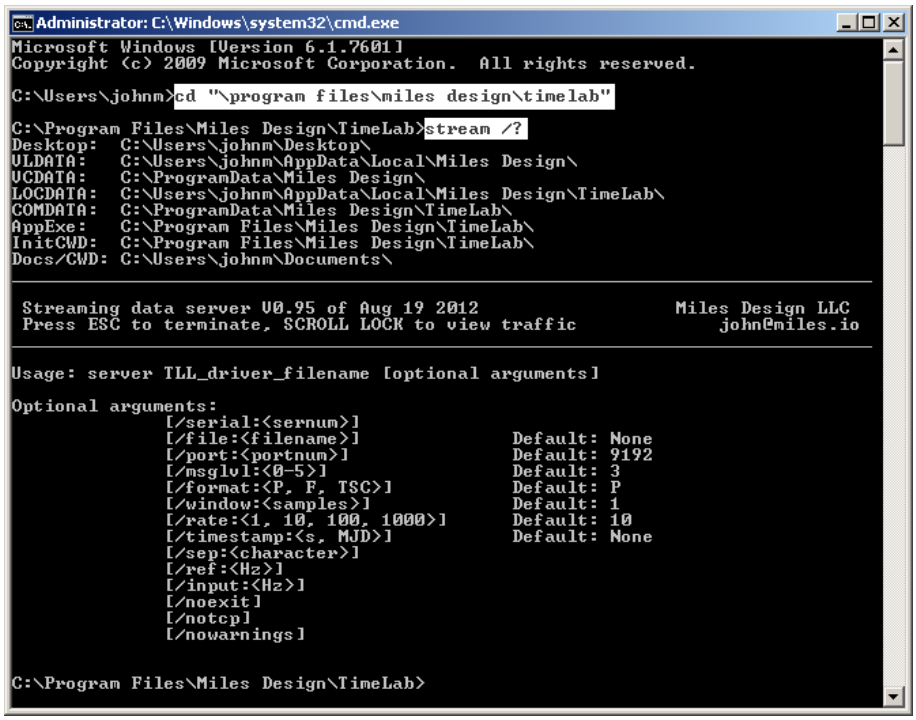
You can launch **STREAM.EXE** from a Windows desktop shortcut, but it's preferable to run it from a command prompt ("DOS box") instead. Because the program's user interface is based on command-line parameters rather than GUI controls, you'll find it easier to work at the command prompt during the familiarization process. After you've determined what options and parameters are needed in your application, you can create a batch file or shortcut to launch **STREAM.EXE** with the same parameters.

STREAM.EXE is installed in the same directory as the rest of the TimeLab package. Unless you specified a different location during installation, you can find it by opening a DOS session and changing the current directory to **c:\Program Files\Miles Design\TimeLab**. To do this, select *Start → Run…* in Windows and enter *cmd* as the name of the program to run. Once the DOS box appears, drag its lower edge to expand the window to a more comfortable size. Then, enter the commands

cd "\program files\miles design\timelab"

stream /?

As seen below, the */?* option causes **STREAM.EXE** to display a brief list of the command-line options available in the current release. (Black on white text indicates typed commands.)



Using STREAM.EXE

Because **STREAM.EXE** performs only phase/frequency stability measurements, its command-line options are relatively limited. Only one parameter is mandatory: you must specify the hardware driver's filename as the first parameter on the command line. Currently, only the TimePod driver is supported, so all **STREAM.EXE** command lines must begin as follows:

stream timepod.tll

Additional options, described below, may be specified as needed.

/serial:<sernum>

When more than one TimePod is connected to the server PC, you may need to use the /serial option to associate a given **STREAM.EXE** instance with a specific instrument. Each instance of **STREAM.EXE** supports only one acquisition at a time, but you can launch as many instances as desired.

Example: *stream timepod.tll /serial:55908.304876*

The command above will launch an instance of **STREAM.EXE** for use with the TimePod whose serial number is 55908.304876. All other parameters will assume their default values, configuring the server to provide phase-difference data on TCP/IP port 9192 at a rate of 10 readings per second.

/file:<filename>

Specifies the name of a file to which the phase-difference or frequency readings will be written. If the specified file already exists, it will be overwritten. Typically the file is shared with other applications that can access it in read-only mode while it's being written by **STREAM.EXE**.

The data written to the file may be 'followed' from within TimeLab by selecting **Acquire→Acquire from live ASCII file**. You can also monitor the data written to the file with a command such as *tail –f <filename>* from Cygwin. Finally, pressing the Scroll Lock key will cause **STREAM.EXE** to print a copy of all lines written to the file to the stdout device (i.e., the DOS console itself).

Example: *stream timepod.tll /notcp /file:%HOMEPATH%\Documents\mydata.txt*

The command above writes a continuous stream of phase-difference readings to the file mydata.txt in the current user's Documents folder. You can then use TimeLab or another application to 'follow' the file. Even if the acquisition fails or otherwise encounters problems, you can use the appropriate **File→Import** option to bring the data into TimeLab at any time.

Additionally, the /notcp option Is used in this example to keep **STREAM.EXE** from attempting to open a network port.

/logfile:<filename>

Specifies the pathname for a logfile which will record status, warning, and error messages. If the /logfile parameter contains an absolute path specification, the log file is placed at that location. Otherwise, if /logfile specifies a filename by itself, the log file is placed in the folder indicated by the first status message written to the Windows console after startup, typically the user's Documents directory.

The default logfile name used when no /logfile parameter is provided is **server.log**.

/port:<portnum>

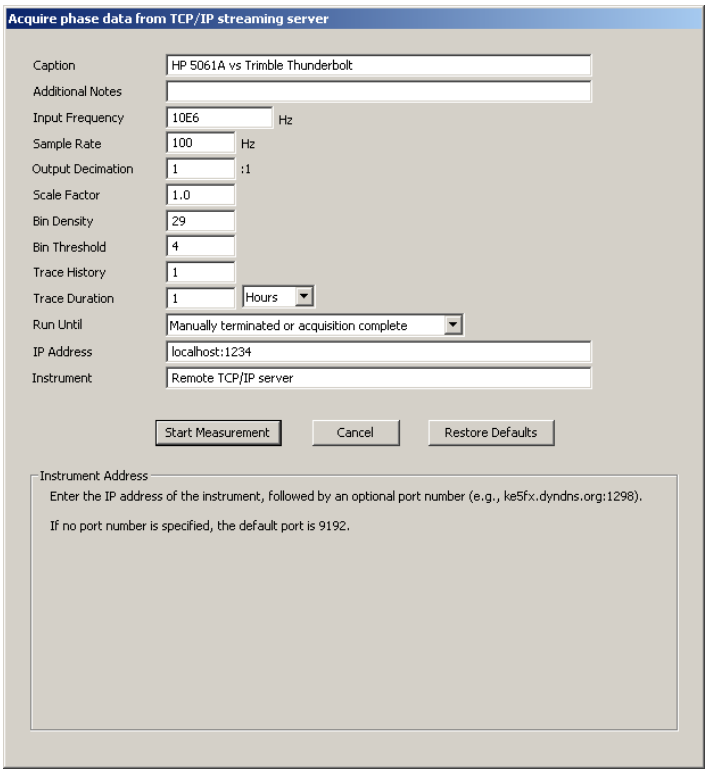
Specifies the TCP/IP port on which **STREAM.EXE** will broadcast incoming phase-difference or frequency readings. By default, **STREAM.EXE** transmits data on port 9192.

To allow remote clients to access **STREAM.EXE**, you may need to configure your firewall or NAT gateway to open this port. Instructions for doing so can be obtained from the equipment manufacturer.

When using software written to capture phase data from the Symmetricom/TSC 5115A/5120A/5125A and compatible instruments, you should specify /port:1298 as well as /format:TSC on the **STREAM.EXE** command line. With TimeLab's **Acquire→Acquire from TCP/IP streaming server** option, only the default phase-difference data format (/format:P) is supported. You can specify any desired port number as long as you also enter it in the acquisition dialog as shown below.

Example: *stream timepod.tll /port:1234 /rate:100*

This command will broadcast phase-difference data on port 1234 at a rate of 100 readings per second. You can receive the data on any machine on your LAN that can access the server, including the server PC itself. In TimeLab, the **Acquire→Acquire from TCP/IP streaming server** option could be used as follows:



/msglvl:<0-5>

Specifies the "verbosity" of informational, warning, and error messages that **STREAM.EXE** will display on the server console. /msglvl:0 will display all available status messages and notices, including internal driver diagnostic messages, while /msglvl:5 will display only fatal error messages. The default message level is 3.

Messages displayed at the server console are also written to the logfile. See the /logfile option for more information.

/format:<P, F, TSC>

Specifies the type of measurement data that will be broadcast via TCP/IP or logged to the specified file.

STREAM.EXE writes phase-difference data by default (/format:P), but you can also record absolute frequency readings with /format:F or TSC 51xx-style phase values with /format:TSC. The latter option emits phase-difference data that has been inverted and scaled by the input frequency.

From TimeLab's perspective, one reason to use **Acquire→Acquire from TCP/IP streaming server** with **STREAM.EXE** instead of the legacy TSC acquisition driver (**Acquire→Symmetricom 5115A / 5120 A / 5125A (Frequency stability)**) is that any warning or error messages from the TimePod driver will be forwarded to the TimeLab

client(s) by **STREAM.EXE** for display in their status lines. In /format:TSC mode only the numeric phase data is transmitted, for compatibility reasons.

See the /input and /ref descriptions below for important additional notes.

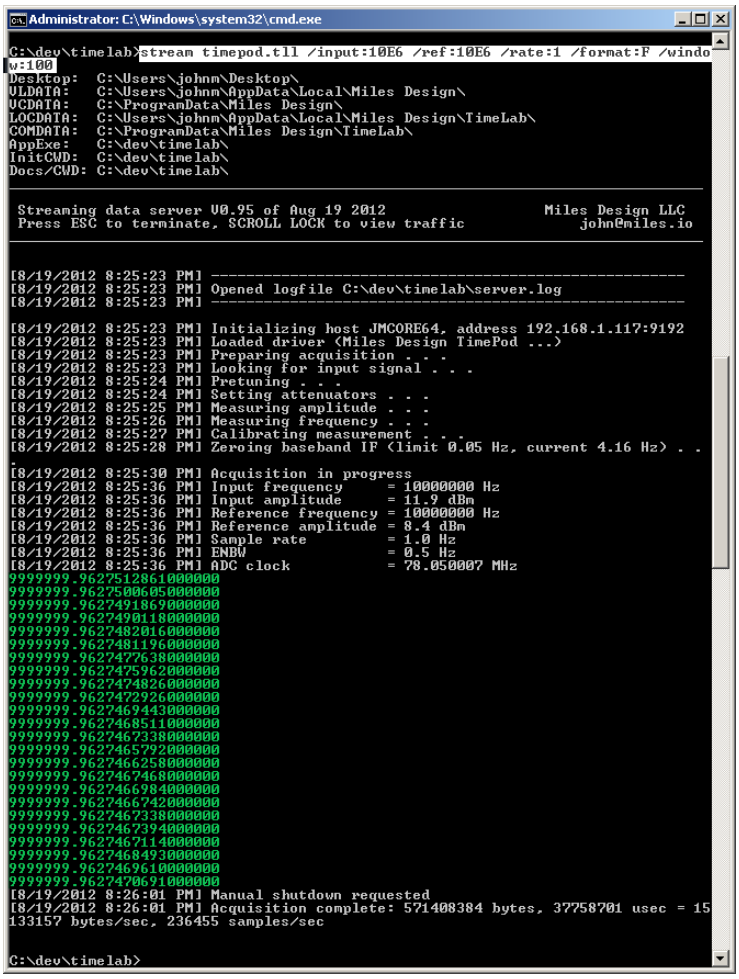
/window:<samples>

When the /format:F option is used to acquire a stream of absolute frequency measurements, each frequency reading is normally derived by subtracting the previous phase-difference reading from each incoming one and dividing the result by the reciprocal of the sampling interval. The resulting frequency difference is applied to the input frequency that was provided explicitly with the /input option or estimated when the measurement began, yielding an absolute frequency in hertz. This value, in turn, is logged to the specified file, sent to network clients via TCP/IP, and/or written to the local console window for diagnostic purposes if the Scroll Lock light is on.

It's possible to achieve much more accurate frequency readings if the frequency-difference value is computed over longer periods of time. The default /window:1 value causes frequency readings to be derived from adjacent pairs of phase-difference readings, as described above. If you specify a larger value with /window:*n*, the frequency differences are computed from phase readings separated by *n* sample intervals (or the number of intervals since the acquisition began, whichever is less.)

In the example below, an attempt was made to measure the actual output frequency from a DDS whose nominal frequency was set to 10 MHz. The use of /rate:1 yielded one reading per second at the minimum supported measurement bandwidth of 0.5 Hz. The DDS clock was derived from the same 10 MHz reference that was supplied to the TimePod, making this a true "residual frequency" measurement.

Example: *stream timepod.tll /input:10E6 /ref:10E6 /rate:1 /format:F /window:100*


Administrator: C:\Windows\system32\cmd.exe
C:\dev\timelab>stream timepod.tll /input:10E6 /ref:10E6 /rate:1 /format:F /windo
w:100
Desktop: C:\Users\johnm\Desktop\
VLDATA: C:\Users\johnm\AppData\Local\Miles Design\
VCDATA: C:\ProgramData\Miles Design\
LOCDATA: C:\Users\johnm\AppData\Local\Miles Design\TimeLab\
COMDATA: C:\ProgramData\Miles Design\TimeLab\
AppExe: C:\dev\timelab\
InitCWD: C:\dev\timelab\
Docs/CWD: C:\dev\timelab\
Streaming data server V0.95 of Aug 19 2012
Miles Design LLC
Press ESC to terminate, SCROLL LOCK to view traffic
john@miles.io
[8/19/2012 8:25:23 PM] Opened logfile C:\dev\timelab\server.log
[8/19/2012 8:25:23 PM] Initializing host JMCORE64, address 192.168.1.117:9192
[8/19/2012 8:25:23 PM] Loaded driver (Miles Design TimePod ...)
[8/19/2012 8:25:23 PM] Preparing acquisition . . .
[8/19/2012 8:25:23 PM] Looking for input signal . . .
[8/19/2012 8:25:24 PM] Pretuning . . .
[8/19/2012 8:25:24 PM] Setting attenuators . . .
[8/19/2012 8:25:25 PM] Measuring amplitude . . .
[8/19/2012 8:25:26 PM] Measuring frequency . . .
[8/19/2012 8:25:27 PM] Calibrating measurement . . .
[8/19/2012 8:25:28 PM] Zeroing baseband IF (limit 0.05 Hz, current 4.16 Hz) . .
.
[8/19/2012 8:25:30 PM] Acquisition in progress
[8/19/2012 8:25:36 PM] Input frequency = 10000000 Hz
[8/19/2012 8:25:36 PM] Input amplitude = 11.9 dBm
[8/19/2012 8:25:36 PM] Reference frequency = 10000000 Hz
[8/19/2012 8:25:36 PM] Reference amplitude = 8.4 dBm
[8/19/2012 8:25:36 PM] Sample rate = 1.0 Hz
[8/19/2012 8:25:36 PM] ENBW = 0.5 Hz
[8/19/2012 8:25:36 PM] ADC clock = 78.050007 MHz
9999999.962751286100000
9999999.962750060500000
9999999.962749186900000
9999999.962749011800000
9999999.962748201600000
9999999.962748119600000
9999999.962747763800000
9999999.962747596200000
9999999.962747482600000
9999999.962747229260000
9999999.962746944300000
9999999.962746851100000
9999999.962746733800000
9999999.962746579200000
9999999.962746625800000
9999999.962746746800000
9999999.962746698400000
9999999.962746674200000
9999999.962746733800000
9999999.962746739400000
9999999.962746711400000
9999999.962746849300000
9999999.962746961000000
9999999.962747069100000
[8/19/2012 8:26:01 PM] Manual shutdown requested
[8/19/2012 8:26:01 PM] Acquisition complete: 571408384 bytes, 37758701 usec = 15
133157 bytes/sec, 236455 samples/sec
C:\dev\timelab>

Here, the frequency readings in green were displayed by turning Scroll Lock on to view the data in real time as it was acquired. After only 24 seconds, the results have converged to within about 3 parts in 10^{-15} of the expected value of 9999999.962747097 MHz. Real-world results will vary, but in any similar test you can expect a trend towards increased accuracy as the frequency-measurement interval grows towards its requested maximum.

See the /input and /ref option descriptions below for important additional notes.

/rate:<1, 10, 100, 1000>

By default, **STREAM.EXE** configures the TimePod driver to make 10 measurements per second. The baseband measurement bandwidth is half of this interval, or 5 Hz by default.

You can use the /rate option to select 1 reading per second to minimize network bandwidth and/or disk space for use with extremely stable sources. Conversely,

measurements of drift-prone sources may require /rate:100 or /rate:1000 to avoid acquisition failures.

Note that measurements made at 1 reading per second are likely to fail if insufficient warmup time is allowed for the DUT, the reference source, or the TimePod itself.

/timestamp:<s, MJD>

/sep:<character>

These related options cause **STREAM.EXE** to timestamp each phase-difference or frequency reading sent to files or network clients.

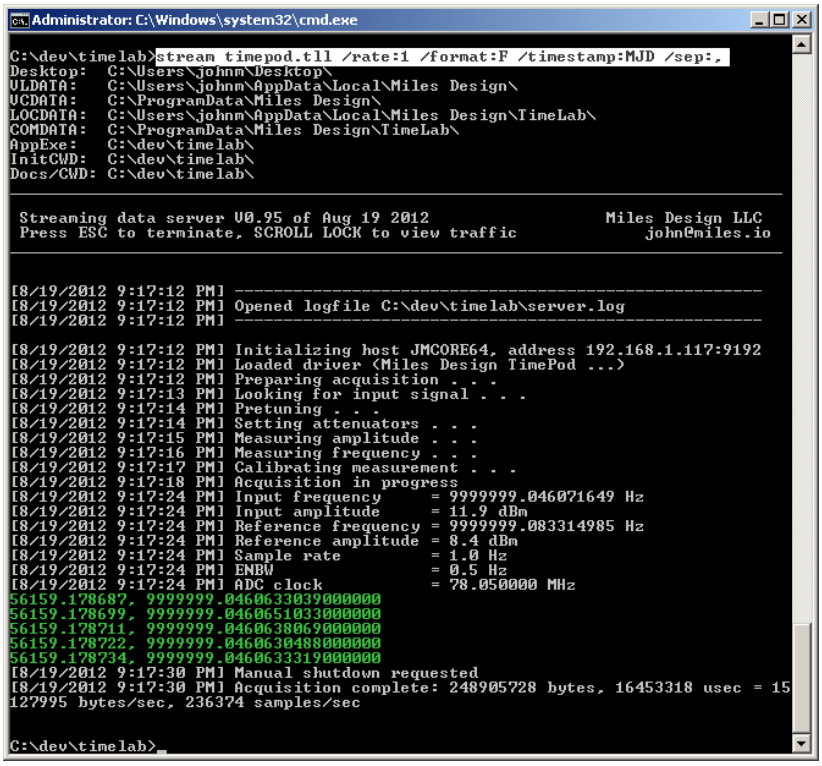
By default, no timestamps are emitted. When /timestamp:MJD is specified, the Modified Julian Date (MJD) associated with each phase-difference or frequency value will be written as the first numeric field on each line, followed by a space character (ASCII 32), then the measurement value itself. MJD timestamps are generated with six digits of decimal precision, resulting in roughly 10 unique MJD values per second.

You can also use /timestamp:s to emit a simple numeric timestamp at one-microsecond precision. These timestamps begin at 0.000000 for the first reading acquired.

In either case, the difference between successive timestamps will always be equal to the reciprocal of the /rate parameter, whose default is 10 readings per second.

The /sep option may be used to specify an additional separator character that will appear before the space between the timestamp and measurement value. If the client requires comma-separated values, for example, /sep:, would be useful. Any numeric value used as the /sep: argument is treated as an ASCII code; e.g. /sep:9 generates a tab character.

Example: *stream timepod.tll /rate:1 /format:F /timestamp:MJD /sep:,*


Administrator: C:\Windows\system32\cmd.exe
C:\dev\timelab>stream timepod.tll /rate:1 /format:F /timestamp:MJD /sep:,
Desktop: C:\Users\johnm\Desktop\
VLDATA: C:\Users\johnm\AppData\Local\Miles Design\
VCDATA: C:\ProgramData\Miles Design\
LOCDATA: C:\Users\johnm\AppData\Local\Miles Design\TimeLab\
COMDATA: C:\ProgramData\Miles Design\TimeLab\
AppExe: C:\dev\timelab\
InitCWD: C:\dev\timelab\
Docs/CWD: C:\dev\timelab\
Streaming data server V0.95 of Aug 19 2012
Miles Design LLC
Press ESC to terminate, SCROLL LOCK to view traffic
john@miles.io
[8/19/2012 9:17:12 PM] Opened logfile C:\dev\timelab\server.log
[8/19/2012 9:17:12 PM] Initializing host JMCORE64, address 192.168.1.117:9192
[8/19/2012 9:17:12 PM] Loaded driver (Miles Design TimePod ...)
[8/19/2012 9:17:12 PM] Preparing acquisition . . .
[8/19/2012 9:17:13 PM] Looking for input signal . . .
[8/19/2012 9:17:14 PM] Pretuning . . .
[8/19/2012 9:17:14 PM] Setting attenuators . . .
[8/19/2012 9:17:15 PM] Measuring amplitude . . .
[8/19/2012 9:17:16 PM] Measuring frequency . . .
[8/19/2012 9:17:17 PM] Calibrating measurement . . .
[8/19/2012 9:17:18 PM] Acquisition in progress
[8/19/2012 9:17:24 PM] Input frequency = 9999999.046071649 Hz
[8/19/2012 9:17:24 PM] Input amplitude = 11.9 dBm
[8/19/2012 9:17:24 PM] Reference frequency = 9999999.083314985 Hz
[8/19/2012 9:17:24 PM] Reference amplitude = 8.4 dBm
[8/19/2012 9:17:24 PM] Sample rate = 1.0 Hz
[8/19/2012 9:17:24 PM] ENBW = 0.5 Hz
[8/19/2012 9:17:24 PM] ADC clock = 78.050000 MHz
56159.178687, 9999999.046063303900000
56159.178699, 9999999.046065103300000
56159.178711, 9999999.046063806900000
56159.178722, 9999999.046063048800000
56159.178734, 9999999.046063331900000
[8/19/2012 9:17:30 PM] Manual shutdown requested
[8/19/2012 9:17:30 PM] Acquisition complete: 248905728 bytes, 16453318 usec = 15
127995 bytes/sec, 236374 samples/sec
C:\dev\timelab>

/ref:<Hz>

/input:<Hz>

When acquiring data directly from a TimePod, TimeLab rounds its reference and input frequency estimates according to the process described on pages 43-45 ("Phase/frequency measurements with the TimePod"). The TimeLab acquisition dialog also allows you to enter explicit input and reference frequencies if desired. As explained in that section of the manual, the reference-rounding process provides the best achievable frequency count chart accuracy when highly-accurate references are in use, while the ability to enter explicit input and reference frequencies allows you to make accurate residual phase measurements.

However, it's important to understand that *no input or reference frequency rounding is performed by STREAM.EXE.* The accuracy of the input and reference frequencies that are displayed at the console window when **STREAM.EXE** is launched depends solely on the accuracy of the TimePod's free-running internal clock.

Consequently, if you wish to make phase-stability measurements without an artificial frequency offset, or if you wish to obtain accurate frequency counts in TimeLab's **Measurement→Frequency difference (f)** view (or when using **STREAM.EXE**'s /format:F mode), you should use the /ref and /input options to specify the nominal reference and input frequencies explicitly. See the preceding demonstration of the /window option for an example.

/autoupdate

The /autoupdate function is useful in scenarios where **STREAM.EXE** may need to be recompiled, upgraded, or otherwise replaced over a network without manual intervention at the server console. When /autoupdate is present on the command line, **STREAM.EXE** will periodically check for the presence of a file called **STREAM.EX1** in the same directory as the executable. If **STREAM.EX1** is ever found, **STREAM.EXE** will terminate with exit(2), reporting the message "Server terminated due to release of new version."

This feature depends on the use of a batch file or script to launch **STREAM.EXE**. Upon detecting exit code 2 when **STREAM.EXE** terminates, the batch file or script should overwrite **STREAM.EXE** with a copy of **STREAM.EX1**, delete **STREAM.EX1**, and then relaunch the new **STREAM.EXE** process with the same set of command-line parameters. When any other exit code is returned by **STREAM.EXE**, the batch file or script can exit normally or take other actions.

Two examples of /autoupdate batch files are included in the TimeLab directory. **SERVE.BAT** is a generic launcher for **STREAM.EXE** that specifies the timepod.tll and /autoupdate options on behalf of the user, passing along any other command-line options as well. **TSC.BAT** demonstrates the use of the /format:TSC and /port:1298 options along with /autoupdate.

SERVE.BAT appears below in its entirety.

```
@echo off

rem
rem Batch file to launch TCP streaming data server with TimePod driver
rem and automatic restart capability
rem

:start

rem
rem If stream.ex1 exists in the directory, rename it to stream.exe and run it
rem (passing any command-line arguments that were originally used with
rem serve.bat)
rem

if not exist stream.ex1 goto serve
copy stream.ex1 stream.exe >nul
del stream.ex1
:serve
if not exist stream.exe goto bail

stream timepod.tll /autoupdate %*

rem
rem If stream server exits with code 2, it means that the /autoupdate option
rem detected a new copy of stream.ex1 in the directory.  Go back and launch
rem the new version without any manual intervention
rem

if errorlevel 2 goto start
:bail
```

/noexit

By default, **STREAM.EXE** will disconnect any attached clients, close its destination file (if any), and terminate with exit(1) under a variety of abnormal conditions, including loss of input or reference signals, out-of-range signal frequencies or amplitude levels, or other acquisition errors. To force the server to restart itself instead of terminating after errors that may be transient or otherwise nonfatal in nature, use the /noexit option.

Attached clients will still be disconnected when the server restarts, but they may reconnect if desired, assuming the server is able to start a new acquisition. If a destination file was specified with the /file option, it will remain open and the new acquisition process will continue to update it.

Note that certain conditions such as memory allocation failures and loss of the TimePod's USB connection are still considered fatal even with /noexit, and will cause **STREAM.EXE** to terminate.

/notcp

You can use /notcp to keep **STREAM.EXE** from opening any network ports or otherwise attempting to act as a TCP/IP server. See the description of the /file option for an example.

/nowarnings

When /nowarnings is used, certain conditions such as excessive reference/input frequency and amplitude drift that would normally result in warning messages will generate error messages instead. Subsequently, **STREAM.EXE** will disconnect any attached clients, close its destination file (if any), and terminate with exit(1).

If /noexit is used in conjunction with /nowarnings, the server will attempt to restart itself under these circumstances rather than terminating. See the /noexit option description for further details.

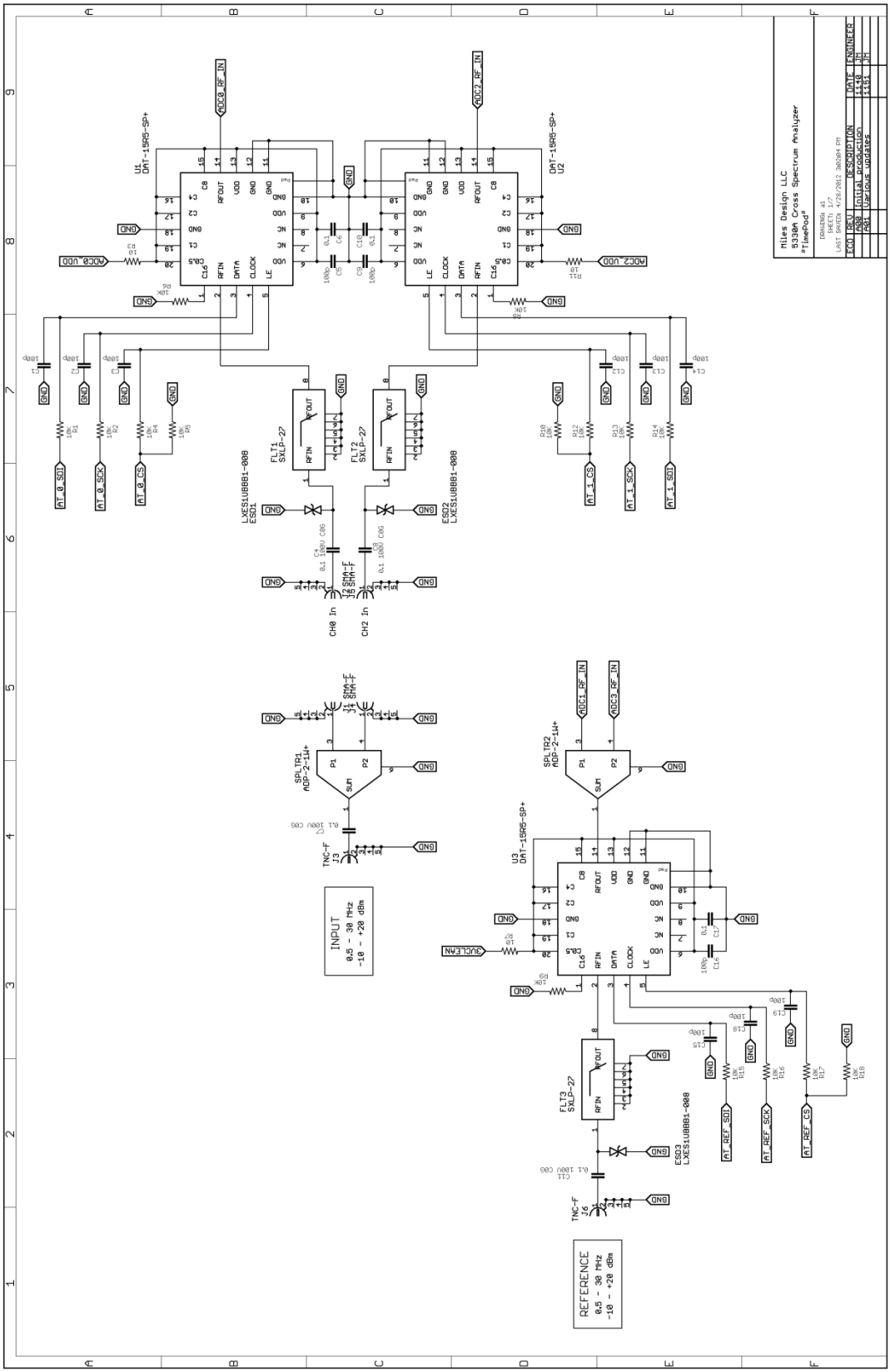
Appendix: Schematic diagrams and service notes

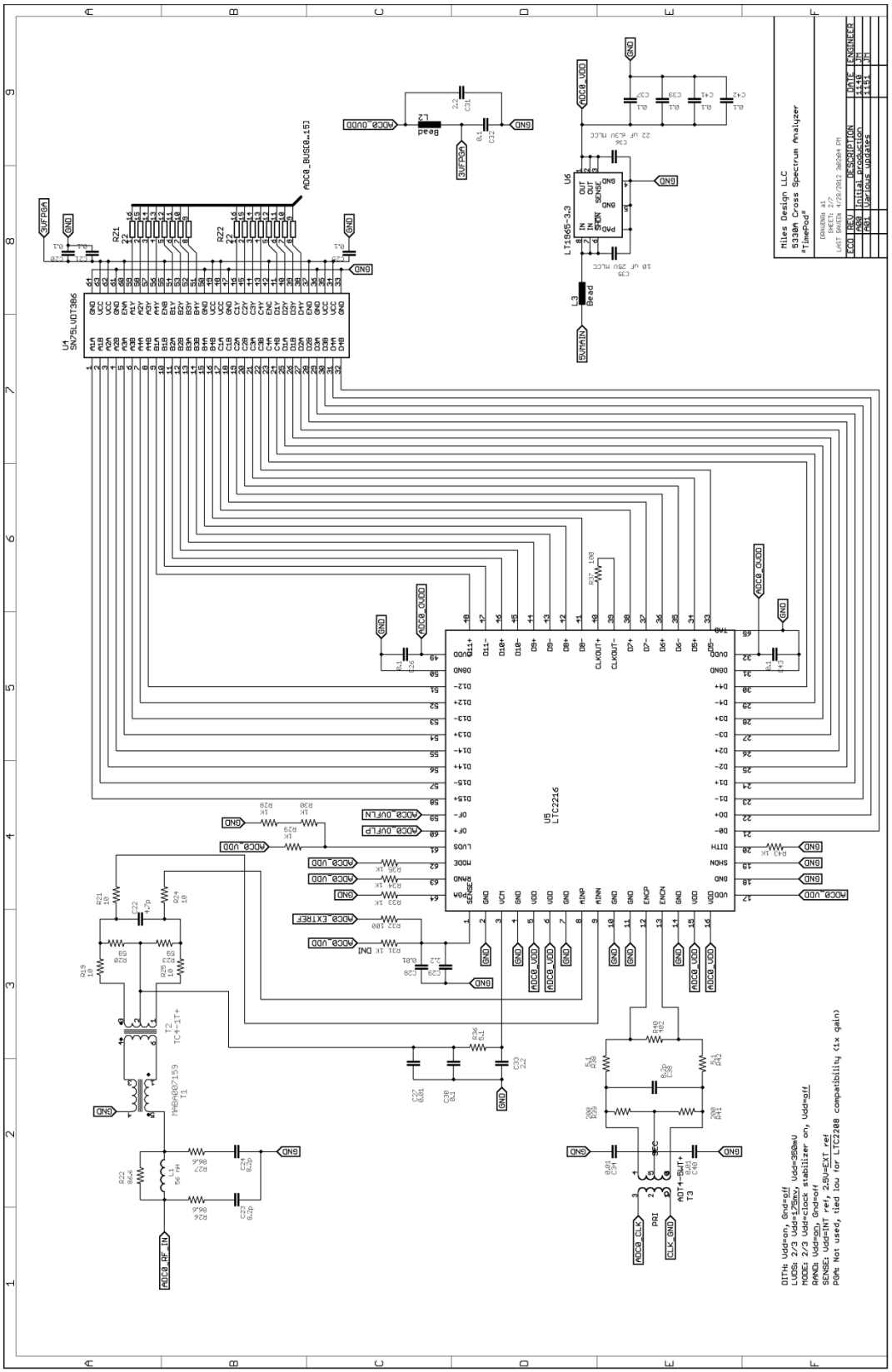
The TimePod® 5330A's FPGA constraints file, *timepod.ucf*, is supplied in the *drivers/TimePod* subdirectory beneath the TimeLab installation folder. The USB interface API and Cypress FX2LP microcontroller source code is also supplied in *XEM3005.zip*. These resources are provided under the terms of the TimeLab software license (page 157) to support the development of custom measurement applications with the TimePod hardware. As stated in the license, no warranties are provided with regard to the use of this information in your own projects.

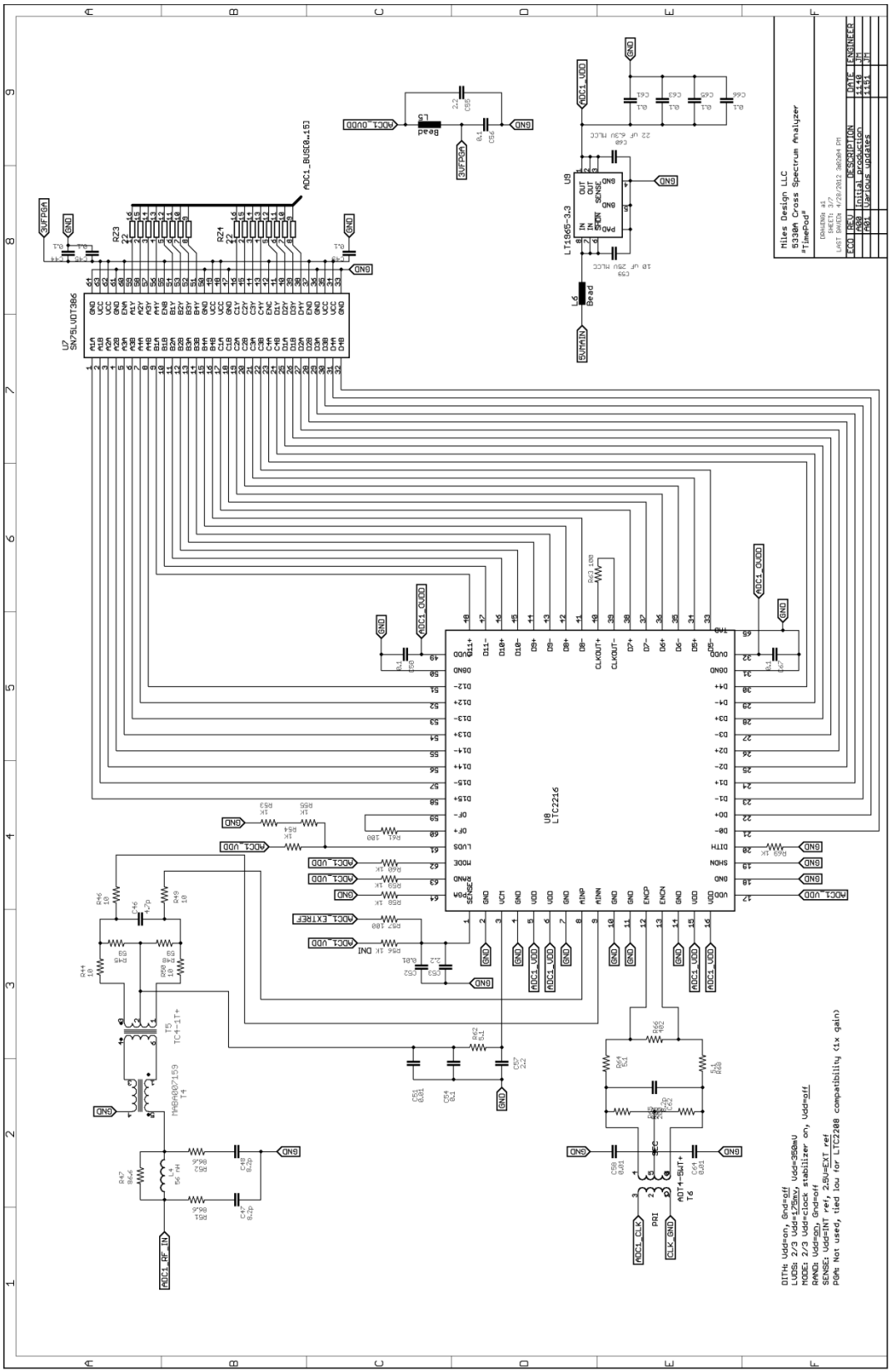
To disassemble the 5330A, first remove the nuts and washers from the TNC jack and SMA jacks on the **INPUT** panel, then remove the eight screws along the **INPUT** panel's perimeter. After the **INPUT** panel has been removed, the instrument's top or bottom cover will slide off after removing two additional screws near the corresponding edge of the **REF IN** panel. Do not unscrew the fasteners that attach the **POWER** or **USB** jacks to the **REF IN** panel unless replacing one of these components.

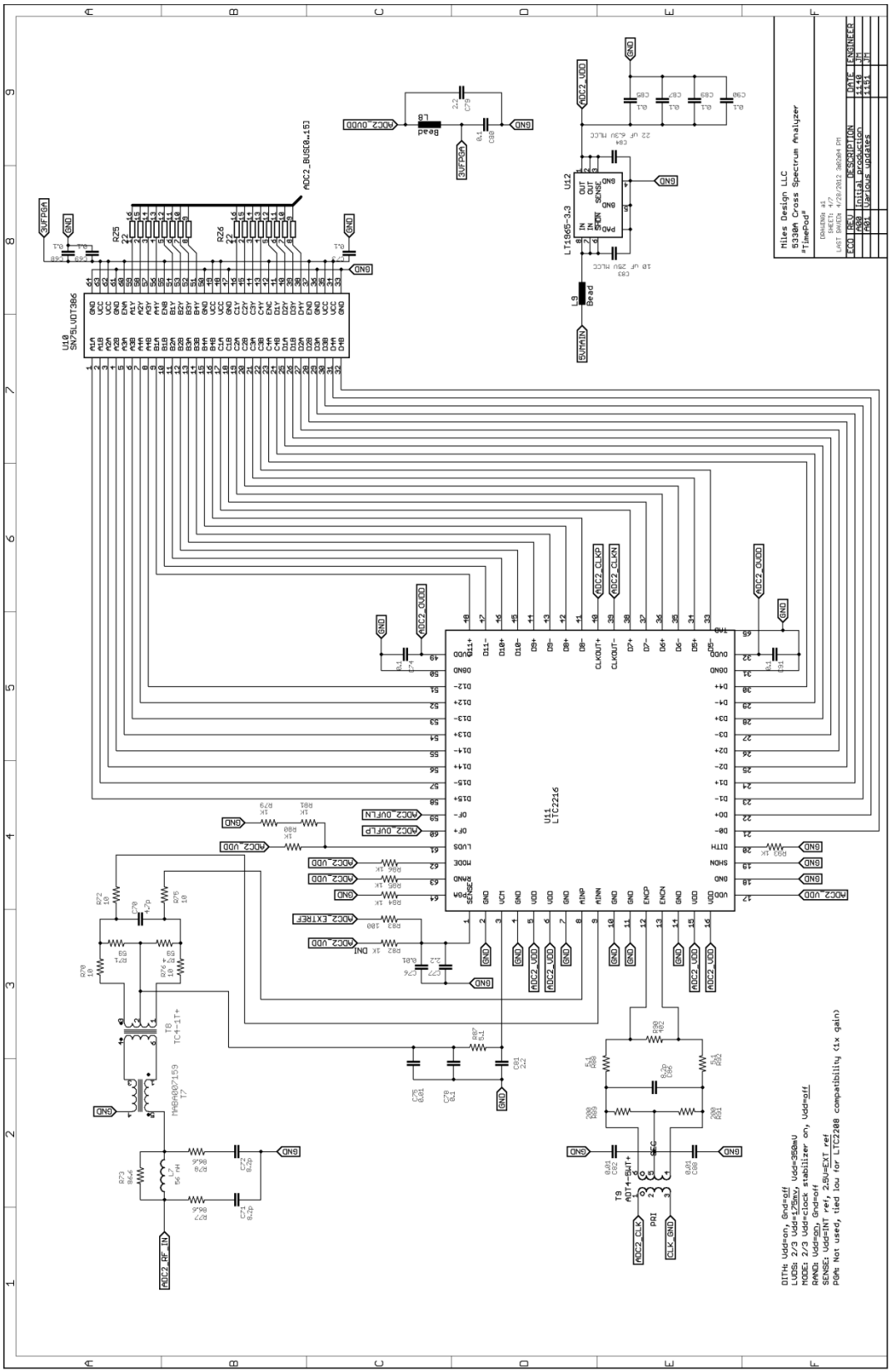
Reassemble the 5330A in the opposite order of disassembly. Prior to installing the **INPUT** panel with its eight mounting screws, carefully position the inner washers and nuts on the four SMA jack barrels so that they will contact the inner surface of the **INPUT** panel under light tension after the mounting screws are tightened. You can either align the washers visually, or temporarily install the **INPUT** panel without the top cover in order to tighten the SMA nuts against the panel's inner surface. *Torque applied to all SMA nuts, both inner and outer, should be the minimum necessary to secure the fasteners*. The goal is to maintain good electrical contact with the **INPUT** panel while minimizing the net mechanical stress on the SMA jacks' solder connections after their outer nuts are tightened.

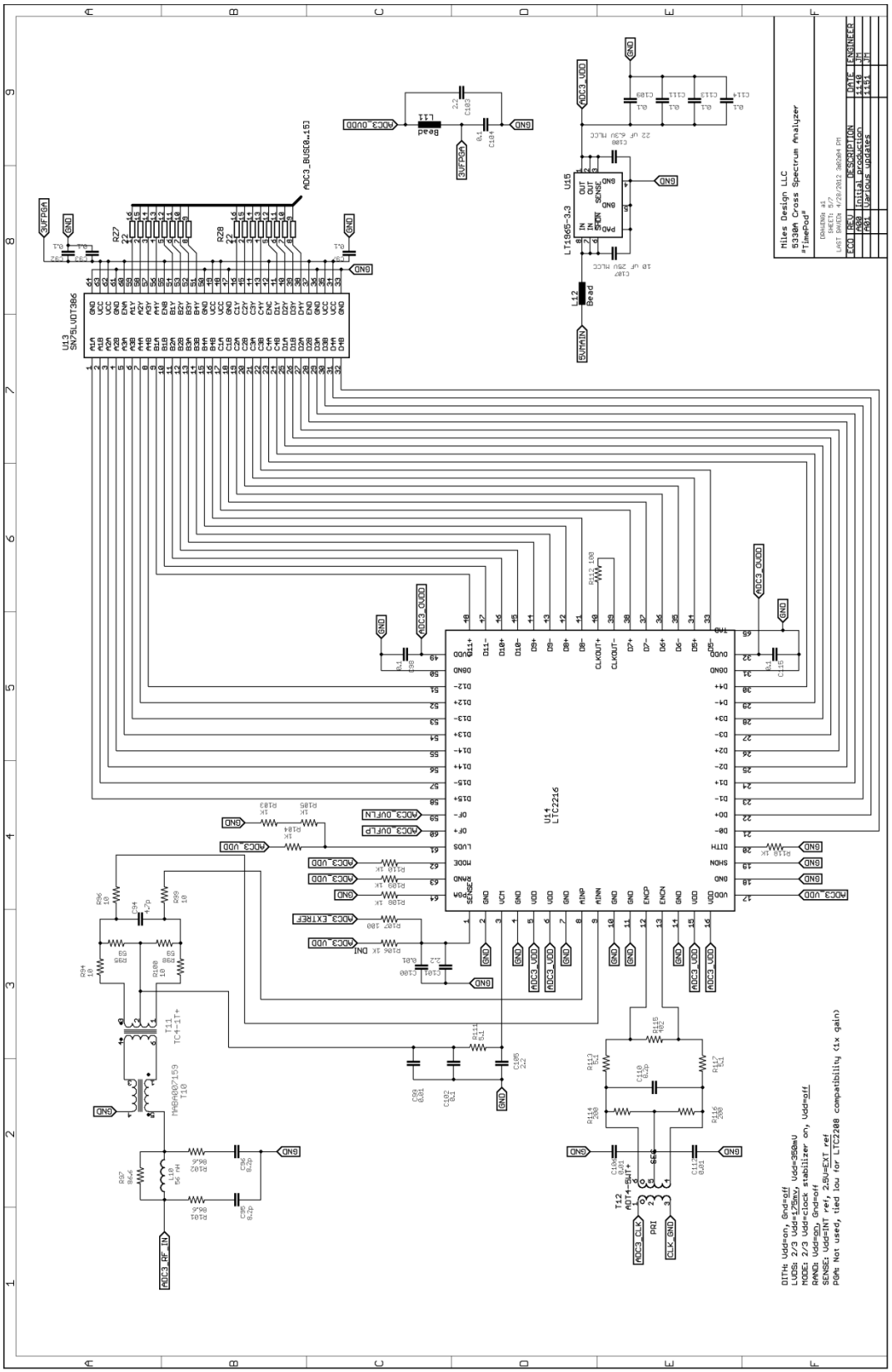
Reinstall the washer and nut on the **INPUT** TNC jack after all other fasteners are in place. As with the SMA nuts, torque should be the minimum necessary to keep the nut secure.

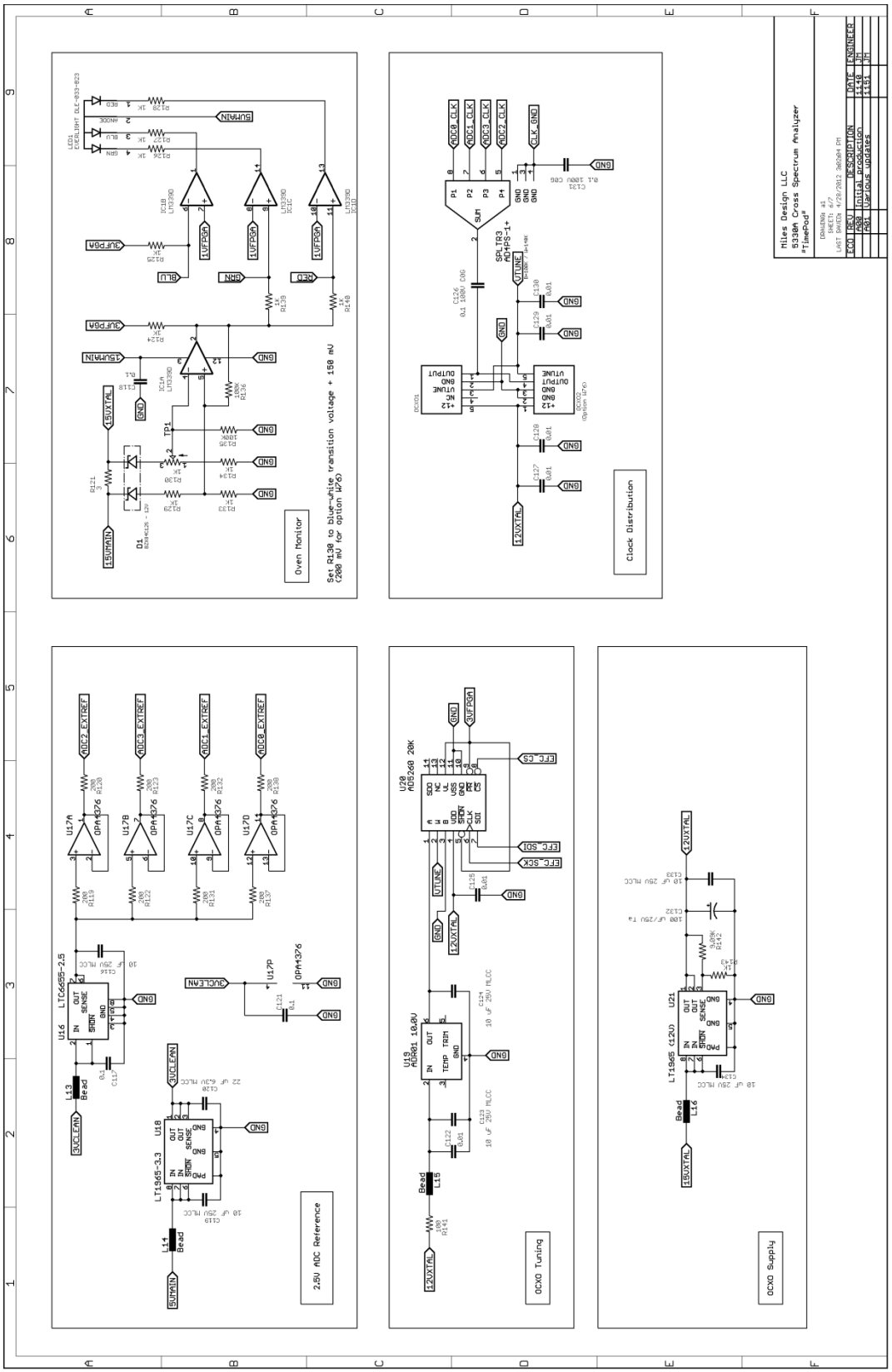

Miles Design LLC
5330A Cross Spectrum Analyzer
"TimePod"
INPUT
0.5 - 30 MHz
-10 - +20 dBm
REFERENCE
0.5 - 30 MHz
-10 - +20 dBm
ADC0_RF_IN
ADC2_RF_IN
ADC1_RF_IN
ADC3_RF_IN
U1 DAT-15R5-SP+
U2 DAT-15R5-SP+
U3 DAT-15R5-SP+
SPLTR1 ADP-2-14+
SPLTR2 ADP-2-14+
FLT1 SXLP-27
FLT2 SXLP-27
FLT3 SXLP-27
ESD1 LXES1LUBB1-008
ESD2 LXES1LUBB1-008
ESD3 LXES1LUBB1-008
J5 SMA-F
J6 SMA-F
J3 TNC-F
J4 SMA-F
AT_0_SDI
AT_0_SCK
AT_0_CS
AT_1_CS
AT_1_SCK
AT_1_SDI
AT_REF_SDI
AT_REF_SCK
AT_REF_CS
3VCLEAN
GND
ADC0_VDD
ADC2_VDD
CH0 In
CH2 In
DRAWING #1
SHEET: 1/7
LAST SAVED: 4/20/2012 2:06:04 PM
ECO | REV | DESCRIPTION | DATE | ENGINEER
ECO | A | Initial production | 11/16 | JH
ECO | A01 | Various updates | 11/51 | JH

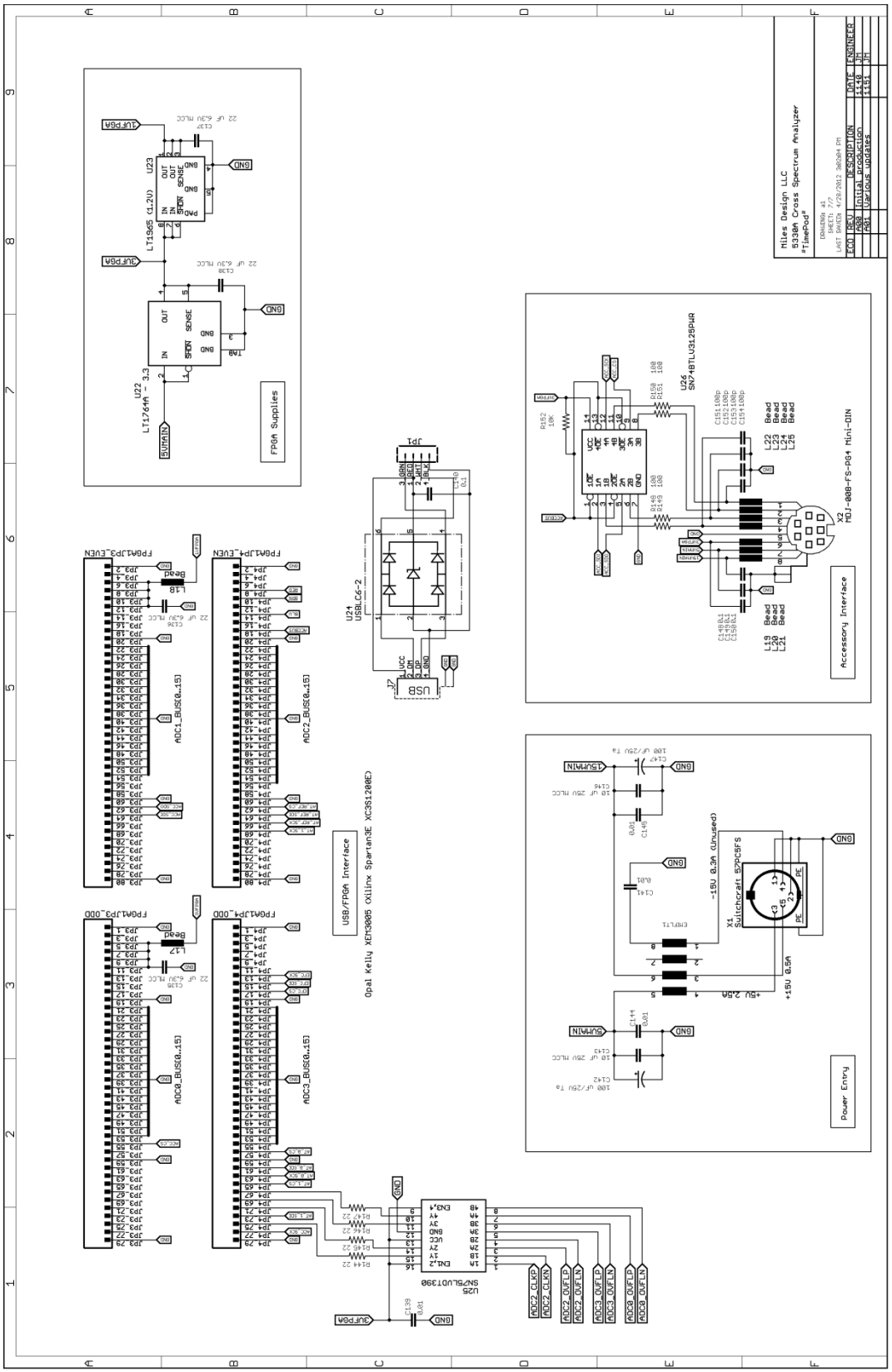

Miles Design LLC
9338A Cross Spectrum Analyzer
"TimePod"
DESCRIPTION
Initial production
Various updates
DITH: Udd=on, Gnd=off
LVDS: 2/3 Udd=1.75mA, Udd=350mV
MODE: 2/3 Udd=clock stabilizer on, Udd=off
RAND: Udd=on, Gnd=off
SENSE: Udd=INT ref, 2.5V=EXT ref
PDN: Not used, tied low for LTC2208 compatibility (1x gain)


Miles Design LLC
9338A Cross Spectrum Analyzer
"TimePod"
U8
LTC2216
U7
SN75LVDT386
U9
LT1965-3.3
T5
TC4-1T+
T6
ADT4-6WT+
T4
MABA007159
ADC1_AF_IN
ADC1_CLK
CLK_GND
ADC1_BUS0..15
3VFPGA
5VMAIN
ADC1_VDD
ADC1_OVDD
GND
DITH: Vdd=on, Gnd=off
LVDS: 2/3 Vdd=1.75mA, Vdd=350mV
MODE: 2/3 Vdd=clock stabilizer on, Vdd=off
RAND: Vdd=on, Gnd=off
SENSE: Vdd=INT ref, 2.5V=EXT ref
PDN: Not used, tied low for LTC2208 compatibility (1x gain)


Miles Design LLC
5330A Cross Spectrum Analyzer
"TimePod"
DESCRIPTION
Initial production
Various updates
DITH: Udd=on, Gnd=off
LVDS: 2/3 Udd=1.75mA, Udd=350mV
MODE: 2/3 Udd=clock stabilizer on, Udd=off
RAND: Udd=on, Gnd=off
SENSE: Udd=INT ref, 2/3=EXT ref
PDN: Not used, tied low for LTC2208 compatibility (1x gain)


Miles Design LLC
9338A Cross Spectrum Analyzer
"TimePod"
DESCRIPTION
Initial production
Various updates
DITH: Udd=on, Gnd=off
LVDS: 2/3 Udd=1.75mA, Udd=350mV
MODE: 2/3 Udd=clock stabilizer on, Udd=off
RAND: Udd=on, Gnd=off
SENSE: Udd=HIN ref, 2/3=EXT ref
PDN: Not used, tied low for LTC2208 compatibility (1x gain)


Miles Design LLC
9338A Cross Spectrum Analyzer
"TimePod"
Oven Monitor
Clock Distribution
2.5V ADC Reference
OCXO Tuning
OCXO Supply


Miles Design LLC
5338A Cross Spectrum Analyzer
"TimePod"
FPGA Supplies
Accessory Interface
Power Entry
USB/FPGA Interface
Opal Kelly XEM3005 (Xilinx Spartan3E XC3S1200E)
SN74BTLV3125PWR
MDJ-008-FS-P04 Mini-DIN
Switchcraft S2PC5FS
SN74SLVDT390
LT1764A-3.3
LT1965 (1.2V)
USB
ADC1_BUS(0..15)
ADC2_BUS(0..15)
ADC0_BUS(0..15)
ADC3_BUS(0..15)
ADC2_CLKP
ADC2_CLKN
ADC2_OUTLP
ADC2_OUTLN
ADC3_OUTLP
ADC3_OUTLN
ADC0_OUTLP
ADC0_OUTLN
+15V 2.5A
+15V 0.5A
-15V 0.3A (unused)

Appendix: Legal and regulatory notices

Federal Communications Commission Statement

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

Changes or modifications not expressly approved by the party responsible for compliance may void the user's authority to operate the equipment.

EC Declaration of Conformity

Manufacturer:	Miles Design LLC, 16758 45th Ave NE, Lake Forest Park, Washington 98155 USA	
Product:	MD 5330A Programmable Cross Spectrum Analyzer	
Description:	Small/portable electronic instrument for laboratory use by qualified personnel, powered by external supply with UL/CUL/TUV/CB/CE approvals	
Directives:	2006/95/EC	The Low Voltage Directive and its amending directives
	2004/108/EC	The Electromagnetic Compatibility Directive and its amending directives
	2002/95/EC	Restrictions on Hazardous Substances and its amending directives
Standards:	EN 61010-1:2001	Safety Requirements for Electrical Equipment for Measurement, Control and Laboratory Use – Part 1: General Requirements
	EN 61326-1:2001	Electrical Requirements for Electrical Equipment for Measurement, Control and Laboratory Use – Part 1: General Requirements
Signature:	I hereby certify that this equipment has been designed and manufactured in accordance with the above referenced Standards, and complies with all essential requirements of the Directives.	

John Miles
Principal, Miles Design LLC
February 24, 2012

Performance Certification and Validation

Miles Design LLC certifies that this product met its published specifications at time of delivery. Because measurements made by the TimePod® 5330A Programmable Cross Spectrum Analyzer are always based on differences between the signals applied to the INPUT and REF IN ports, no specific periodic calibration procedures are required. In the absence of errors reported by the measurement software, no specific maintenance procedures are necessary, including preventative or periodic maintenance.

For formal verification purposes, the procedures outlined in the default mask definition file (**masks.txt**) will permit customer verification of all performance specifications using commonly available test signals. For more information, select **Masks→Edit mask definitions** in TimeLab to open the mask definition file and refer to the *TimePod 5330A performance test masks* section. The automated test script **5330A_performance_test.js** may also be used to verify all performance specifications.

Limited Warranty

Miles Design LLC warrants the TimePod® 5330A Programmable Cross Spectrum Analyzer against all defects in materials and workmanship for a period of ONE (1) year from the date of original customer delivery. The sole responsibility of Miles Design LLC under this Warranty is limited to repair or replacement of the TimePod® 5330A Programmable Cross Spectrum Analyzer, at the sole discretion of Miles Design LLC. Contact information for Miles Design LLC for technical support and warranty claim purposes may be found on the inside of the first (cover) page of this manual.

The material contained in this document is provided “as is,” and is subject to change without notice in future editions. Further, to the maximum extent permitted by applicable law, Miles Design disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Miles Design shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Miles Design and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

THE REMEDIES PROVIDED HEREIN ARE THE CUSTOMER'S SOLE AND EXCLUSIVE REMEDIES. MILES DESIGN LLC SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, STRICT LIABILITY, OR TORT, ARISING FROM THE USE OF THIS EQUIPMENT, ITS ACCOMPANYING SOFTWARE, OR ITS ACCOMPANYING DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software License: TimeLab

http://www.miles.io/timelab/readme.htm

Copyright (c) 2011-2012 Miles Design LLC. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted, provided that the following conditions are met:

1. For all components that include source code provided by Miles Design LLC, redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Names and trademarks associated with Miles Design LLC may not be used to endorse or promote products derived from this software without specific prior written permission from Miles Design LLC.

THIS SOFTWARE IS PROVIDED BY MILES DESIGN LLC ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL MILES DESIGN LLC BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software License: FFTSS

http://www.ssisc.org/fftss/index.html.en

Copyright 2002-2007 Akira Nukada. All rights reserved.
Copyright 2002-2007 The SSI Project. All rights reserved.
The Scalable Infrastructure Project, supported by "Development of Software Infrastructure for Large Scale Scientific Simulation" Team, CREST, JST.
Akira Nishida, Department of Computer Science, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE SCALABLE SOFTWARE INFRASTRUCTURE PROJECT ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE SCALABLE SOFTWARE INFRASTRUCTURE PROJECT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software License: FIDLIB

http://uazu.net

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you

receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library

or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Software License: V8 JavaScript Engine

http://code.google.com/p/v8/

Copyright 2006-2012, the V8 project authors. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE